Ingeniera de Sistemas Tic

Trabajo Final de Grado

EMULADOR DE BICICLETA

Albert Garcia Lagares

Dirigido por Francisco del Águila

14 de Octubre de 2015

AGRADECIMIENTOS

Quiero agradecer a todas las personas que han hecho posible este proyecto final de grado.

Primeramente a mi director de proyecto Francisco del Águila por su ayuda durante el proceso.

También quiero agradecer el apoyo moral, la ayuda durante la realización del proyecto o la ayuda económica a Tania Moriana, Sara Garcia, Juan Garcia y Rosa Mª Lagares.

ABSTRACT

The communication and information technology are increasingly in our lives, this expansion is very big and covers many aspects of our society. Actually all big companies have departments of TIC technologies and develop their own products.

In the sport we can find many products of TIC with high social acceptance. Most athletes train and study themselves with this kind of sistems that collect and process information, and they use it for improve and learn faster and better than ever before. Using sensors and all this information, we can study all kind of sport situations and get better results.

Cycling is not a exception, every day we have more developed systems to get better control on training. The information of the workouts are great opportunities to improve, and also let us create the training simulations. Even a complicated ones simulating hard stages of the best competitions.

Based in this concept, this project offers another possibility for training, a complete system for recording information of the trainings and an emulation of this in a static site with a good interfaces.

Índice

1. Introducción y objetivos

1.1. Introducción

Este proyecto se centra en el uso de las Tecnologías TIC para crear un emulador de entrenamiento de bicicleta. Actualmente, hay muchos ciclistas que practican éste deporte con mucha frecuencia. Éstos están expuestos al mal clima en muchas ocasiones y por eso substituyen el entrenamiento en la calle por sistemas de entrenamiento estáticos (bicicletas estáticas o sistemas acoplados a su bicicleta de entrenamiento). Los entrenamientos de bicicleta en espacios cerrados son muy poco gratificantes y por eso este proyecto soluciona la falta de motivación en estos entrenamientos. Dicho proyecto consta de dos partes: la primera es la obtención de datos y la segunda la emulación de entrenamiento.

Cada vez más existen dispositivos capaces de de emular entrenamientos simulando situaciones reales. En la actualidad existen dos marcas comerciales las cuales ya tienen dispositivos similares al descrito. Estos dispositivos son rodillos de entrenamiento que consisten en poner una bicicleta sobre un mecanismo que frena más o menos el neumático trasero, dependiendo de los entrenamientos predefinidos que nos ofrecen por Internet, éste es la principal limitación de estos sistemas: la necesidad de descargar de Internet rutas predefinidas. Alguno de estos dispositivos ya empiezan a desarrollar sistemas para relacionar un vídeo creado por un usuario con una configuración manual posterior.

Este proyecto pretende crear la forma de registrar rutas con vídeos hechos por uno mismo, sin tener que recurrir a rutas predefinidas por el fabricante o configurar-las manualmente. Además éste dispositivo es totalmente local ofreciendo privacidad.

1.2. Objetivos

El objetivo de este proyecto es conseguir una emulación de entrenamiento lo más parecida a la realidad. Para conseguirlo se tiene que estudiar como se consiguen los datos y como se puede crear un emulador que los utilice de una forma lo más precisa posible.

Objetivos de la obtención de datos del entrenamiento

- Investigar como crear una aplicación Android.
- Ofrecer una interfaz gráfica al usuario fácil de usar y comprender.
- Generar un historial de rutas con todos sus datos.
- Registrar las inclinaciones que va experimentando el usuario durante su recorrido.
- Visualizar la ruta en un mapa, marcando el recorrido hecho.
- Poder gestionar toda la información desde la propia aplicación.
- Obtener datos gps, distancia, tiempo, velocidad y vídeo de la ruta que el usuario quiere posteriormente simular.

Objetivos del emulador

- Diseñar un dispositivo con una interfaz gráfica para el usuario.
- Ofrecer una plataforma de conexión.
- Poder controlar unos rodillos de entrenamiento sincronizándolo con la emulación de la ruta de entrenamiento.
- Visualizar el vídeo acorde a la ruta registrada.
- Gestionar una base de datos de todas las rutas registradas.
- Conseguir un emulador que se ajuste al máximo a la realidad de las rutas realizadas.

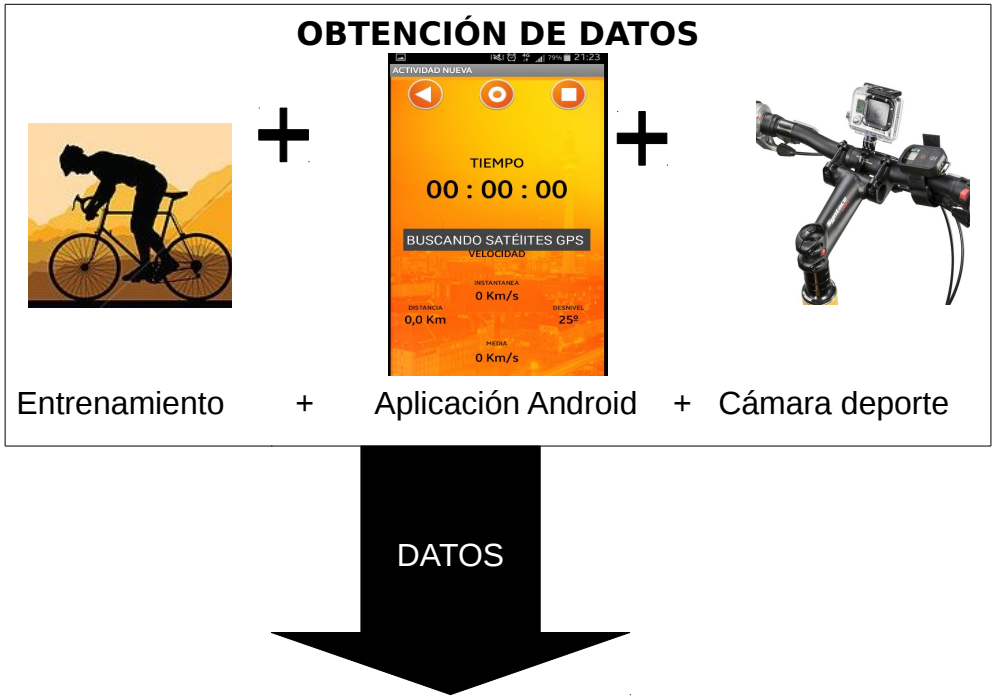
2. Procedimiento

La realización de este proyecto ha realizado siguiendo estos pasos:

1. Investigar los métodos para crear y compilar una aplicación Android.
2. Una vez decidido crear la aplicación con App Inventor, se ha estudiado los diferentes sensores que se necesitaban para poder valorar su efectividad y estudiar el método de usarlos.
3. Una vez decidido como usaremos los sensores de la aplicación y los métodos de cálculo ya podemos empezar a crear la aplicación definitiva, incorporando el factor cámara de deporte.
4. Con la aplicación creada ya tenemos los datos necesarios para la emulación, el siguiente punto es estudiar como hacer el emulador.
5. Estudiando las necesidades y los requisitos mínimos que queremos conseguir, empezamos a crear nuestra aplicación web.
6. Instalamos en una Raspberry el servidor apache, dhcp y hostapd. Con estos puntos se consigue un servidor web con acceso local mediante una red Wifi creada por esta.
7. Desarrollamos la aplicación web. Se ha investigado sobre lenguajes posibles, su funciones que necesitaremos y las desventajas que ofrecen.
8. Con la aplicación Web funcionando, se ha instalado el servomotor y se ha probado su funcionamiento,
9. Con el proyecto al completo funcionando se ha valorado el resultado y las posibles mejoras que puede tener.

3. Arquitectura del proyecto

El proyecto se divide en dos grandes partes: la obtención de datos y la posterior emulación de estos. Para lograr este objetivo es necesaria una aplicación para dispositivos con sistema operativo Android y un emulador, compuesto por un servidor web con conexión Wifi y un control del sistema de entrenamiento estático. La aplicación genera y guarda la información, a la vez que se sincroniza con una cámara de deporte para obtener un vídeo mientras el usuario realiza un entrenamiento. El emulador gestiona los datos y emula el entrenamiento realizado con anterioridad.



3.1. Aplicación Android

Para la obtención de datos se ha diseñado una aplicación para Android que se sincroniza con una cámara de deporte (GoPro 3 White). La función principal de dicha aplicación es generar un documento con toda la información del entrenamiento realizado a la vez que la cámara registra un vídeo del entrenamiento.

Esta aplicación debe ofrecer también al usuario una gestión de fácil uso y compensión para poder visualizar los datos de cada uno de los entrenamientos registrados (distancia, tiempo, recorrido) y poder borrar los entrenamientos deseados. Para realizar esta aplicación se ha optado por buscar una solución que ofreciera mucha rapidez de creación, esta decisión se a tomado porque crear una aplicación que se capaz de cumplir con todos los requisitos no es nada sencillo. Sabiendo que la aplicación es solo una de las dos grandes partes del proyecto, necesitaba encontrar una forma rápida, por eso, después de valorar varias opciones (Eclipse o Android studio), llegue a la conclusión que la única forma viable de crear esta aplicación Android era con MIT App Inventor 2.

MIT App Inventor 2

Es un entorno de desarrollo de aplicaciones para dispositivos Android. Para desarrollar aplicaciones con App Inventor sólo se necesita un navegador web. Esta herramienta de desarrollo es visual y muy fácil de usar. Se constituye de dos partes; App Inventor Designer y App Inventor Blocks Editor.

- App Inventor Designer. En Designer se construye la interfaz de usuario, eligiendo y situando los elementos con los que interaccionará el usuario y los componentes que utilizará la aplicación.

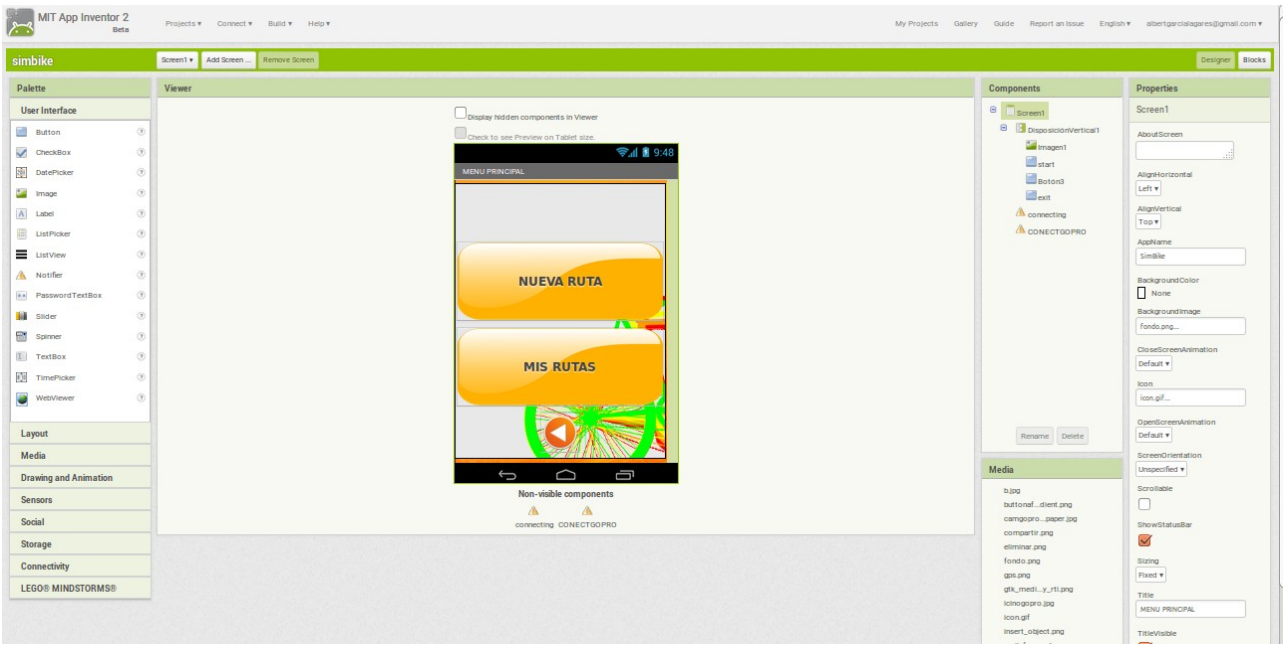


Figura 1: App Inventor Designer

- App Inventor Blocks Editor. En el Blocks Editor se define el comportamiento de los componentes de la aplicación. Su utilización es intuitiva por el método que tiene: el uso de bloques.

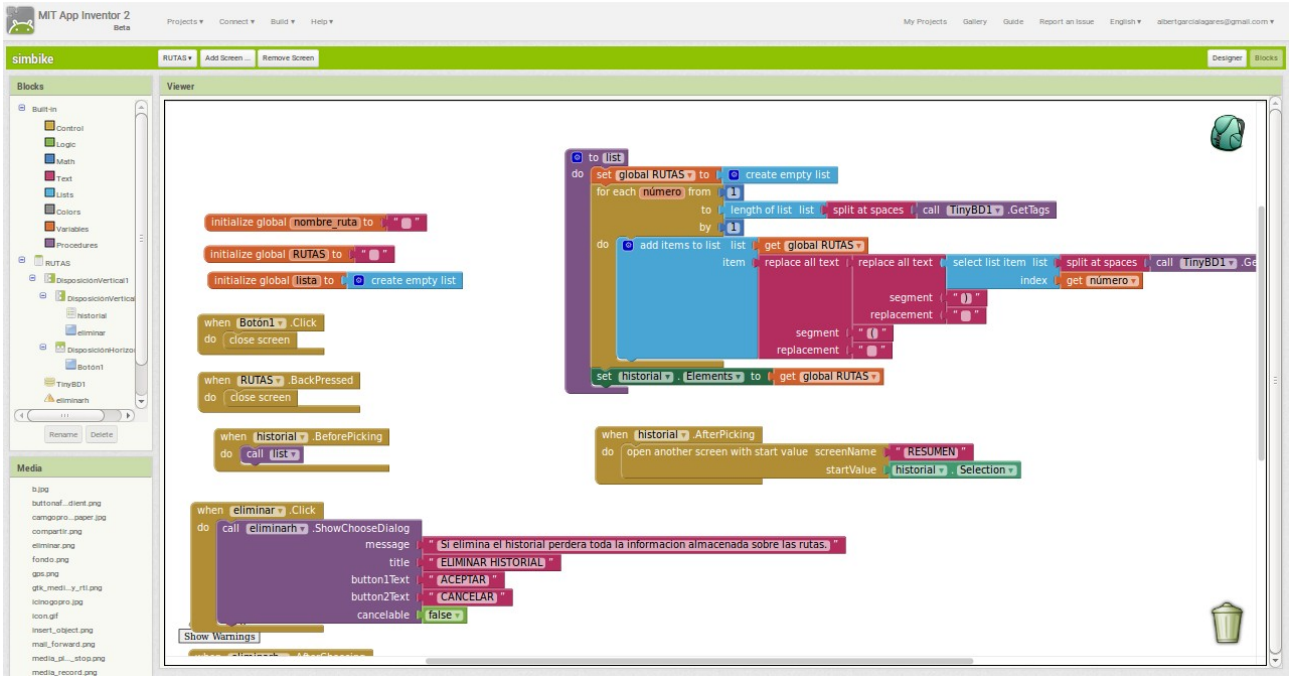


Figura 2: App Inventor Blocks

3.2. Emulador de entrenamiento

Para ofrecer todos los objetivos que se quieren dar al usuario se ha diseñado un dispositivo totalmente independiente con una Raspberry pi B, esta alberga un servidor Web para ofrecernos una aplicación Web. La comunicación con el servidor es a nivel local, con un Wifi generado por el mismo. Además la Raspberry , aprovechando los pins GPIO que este minicomputador ofrece, controla un servomotor. El servomotor actúa mecánicamente sobre unos rodillos de entrenamiento, con el fin de aumentar o disminuir la dificultad del entrenamiento.

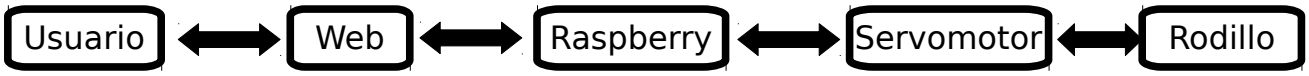


Figura 3: Bloques emulador

En el emulador se distinguen dos procesos claramente diferenciados: la subida de datos al servidor y la emulación de entrenamiento.

Subida de datos al servidor

Una vez el usuario tenga una ruta registrada, tendrá que conectarse al servidor mediante el Wifi y transferir los datos y el vídeo. La aplicación genera un documento con extensión ".txt" ubicado en nuestro dispositivo y la cámara genera un archivo de vídeo albergado en la propia cámara. Con estos dos documentos en un mismo dispositivo, el usuario se conecta a la red wifi generada por el emulador y mediante la página web servida por este y transferirá los dos archivos.

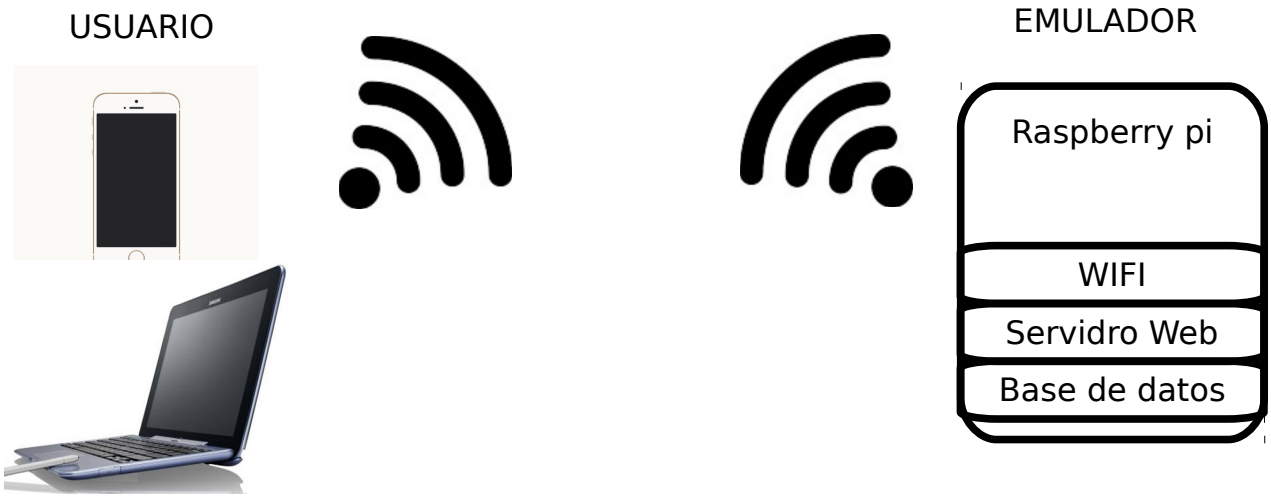


Figura 4: Subida de datos al servidor

Emulación de entrenamiento

Una vez el emulador ya tenga alguna ruta guardada ya podremos emularla, para ello el usuario se conectará con cualquier dispositivo que contenga un navegador web (smartphone, tablet, pc, smartTV, etc.) y mediante la web podrá elegir la ruta a entrenar, gestionarlas y emularlas. El dispositivo de emulación responderá a las ordenes de la web mostrando el vídeo en la propia web y actuando sobre el sistema de entrenamiento (rodillo), seleccionando los niveles de resistencia en función a los datos de la ruta elegida.

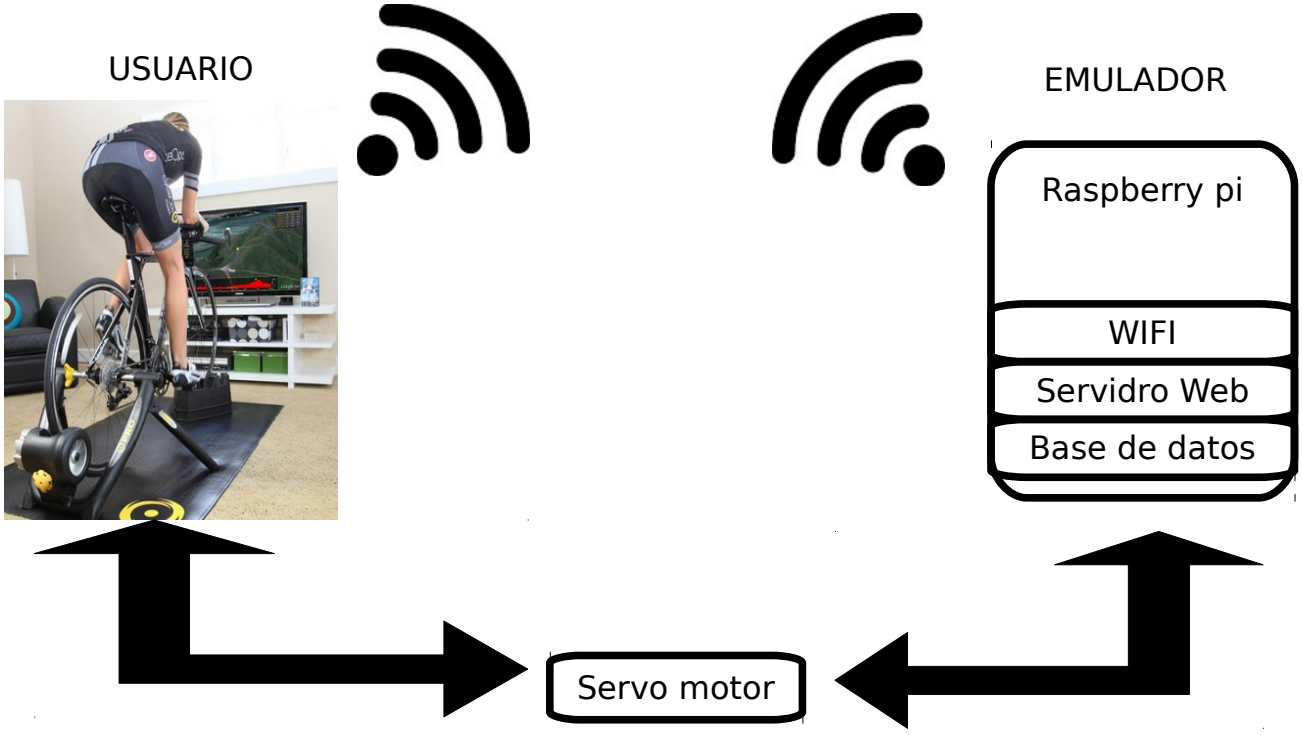


Figura 5: Emulación de un entrenamiento

4. Aplicación Android

Esta aplicación, con nombre “SimBike”, esta creada con MIT Appinventor 2 y necesita de la cámara de deporte GoPro3. La aplicación creada, con el nombre “SimBike”, es capaz de registrar entrenamientos y gestionar una base de datos de éstos. La interfaz gráfica ésta diseñada para facilitar la comprensión y el uso por parte del usuario, se basa en cuatro pantallas muy simples.

La aplicación le sirve al usuario para registrar una ruta y para ver el historial de las rutas guardadas y las informaciones referentes a éstas. A continuación se muestran las pantallas y los flujos entre ellas.

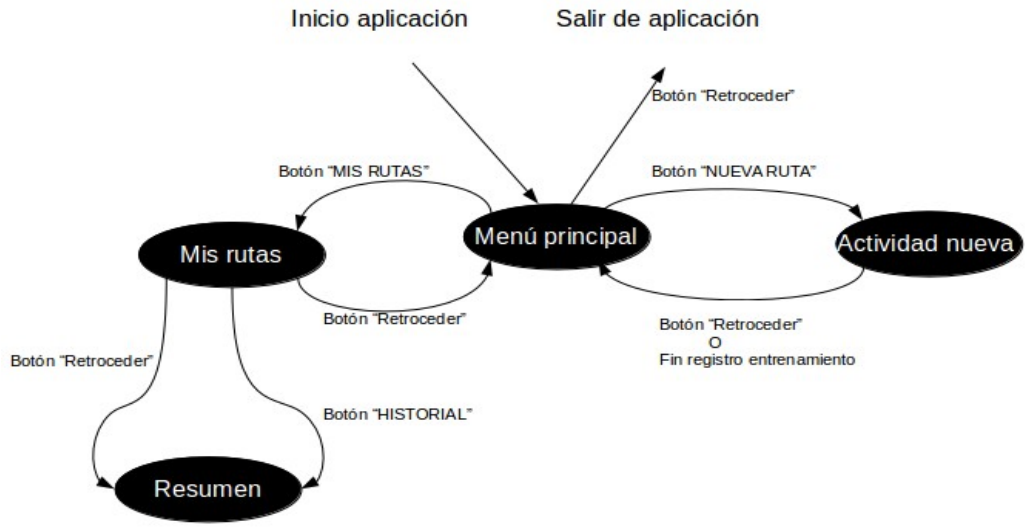


Figura 6: Pantalla “Menú principal”

Los elementos importantes de MIT Appinventor 2 que se han utilizado en esta aplicación son los siguientes:

- Sensor de Ubicación: es un componente no visible que ofrece información sobre la ubicación, incluyendo la longitud, latitud, altitud y dirección. Para que este elemento funcione tenemos que habilitarlo, y debe ser viable determinar la ubicación habilitando en el dispositivo la función GPS. Es posible que la información sobre la ubicación no esté disponible en el momento de iniciar la aplicación. Normalmente hay que esperar un poco para que el proveedor de la localización la encuentre y la utilice, por eso nuestra aplicación “SimBike” se asegura a establecer conexión, espera a iniciar un registro de entrenamiento cuando el sensor de Ubicación ya tiene valores.

- Reloj: es un componente no visible, que permite utilizar un reloj, o un temporizador, y hacer cálculos de tiempos. En nuestra aplicación se usa para hacer el cálculo de tiempo del entrenamiento y como reloj para realizar los cálculos de variables.

- Sensor de Orientación: es un componente no visible que proporciona información acerca de la orientación física del dispositivo en tres dimensiones, la dimensión que nosotros utilizaremos es el Cabeceo. El Cabeceo nos muestra 0º cuando el dispositivo está en horizontal, aumenta hasta 90º cuando la parte superior del dispositivo está apuntando hacia abajo, y hasta 180º cuando está dado la vuelta. Cuando la parte inferior del dispositivo está apuntando al suelo, el Cabeceo disminuye hasta -90º, y sigue disminuyendo hasta -180º cuando se gira completamente.

- Base de datos, TinyDB: es un componente no visible que puede almacenar datos. Las aplicaciones creadas con App Inventor se inicializan cada vez que se ejecutan: si una aplicación inicializa el valor de una variable y el usuario termina la ejecución de la aplicación, el valor de esa variable se perderá, y no estará disponible la próxima vez que se ejecute la aplicación. TinyDB es un almacén de datos persistente para la aplicación, lo que significa que los datos estarán disponibles cada vez que se abra la aplicación.

  Los datos son cadenas de texto que se almacenan en *etiquetas*. Para almacenar datos, se debe de especificar la etiqueta que corresponde a ese dato. Posteriormente, los datos se pueden recuperar con esa misma etiqueta. Solamente existe una base de datos para cada aplicación. Aunque se utilicen distintas TinyDBs en una aplicación, todas ellas apuntan al mismo almacén. Se deben utilizar distintas etiquetas para guardar diferentes datos, y no usar diferentes almacenes.

  Nuestra base de datos tiene como etiqueta el nombre de la ruta y los datos es una lista de strings. Ésta lista contiene cuatro variables: distancia total de la ruta, tiempo total de la ruta y las otras dos es parte del código html guardado durante el registro de la ruta, que posteriormente usaremos para poder visualizar la ruta en la Api de Google Maps en un visor web.

  Estructura de la base de datos:

  (nombre_ruta, [distancia, tiempo_en_formato_string , valores_googleapi1, valores_googleapi2, tiempo_en_segundos])

- Archivo: componente no visible que sirve para guardar o recuperar archivos. Éste componente sirve para escribir o leer archivos en el dispositivo. Inicialmente sirve para escribir archivos en el directorio privado asociado a la aplicación, en el directorio “/sdcard/AppInventor/data”. Para evitar poder tener acceso al archivo donde guardamos la información de la ruta, el cual tendremos que extraer, escribimos el archivos empezando con una barra invertida “/”, así el archivo se escribirá en el directorio “/sdcard”.

- Visor Web: es un componente para ver páginas Web. Se puede especificar la página de inicio desde el Diseñador, o en el Editor de Bloques. Éste componente en nuestra aplicación tiene dos funcionalidades:
    - Mostrar la Api de Google Maps
    - Enviar instrucciones con protocolo http a la cámara de deporte GoPro3.

Registro de entrenamiento

A continuación se van a definir los pasos que tiene que seguir el usuario para registrar un nuevo entrenamiento:

1. El usuario inicia la aplicación para el registro de un entrenamiento, previamente debe activar el sensor GPS y el Wifi. La aplicación cuando inicia muestra un mensaje de aviso para que el usuario active estos servicios. El Wifi se tiene que conectar a la red que nos generará nuestra cámara deportiva GoPro, de la que tenemos que activar su Wifi con el botón lateral del que dispone la cámara.

2. Entramos en la pantalla “NUEVA RUTA”, al iniciarse la aplicación nos pide un nombre para la ruta, el cual no puede estar vacío.

3. Cuando ya estamos en la pantalla “NUEVA RUTA” podremos observar como se sincroniza la cámara de deporte y como el sensor GPS intenta establecer localización. Mientras esto sucede la aplicación nos muestra los ángulos en que se encuentra nuestro dispositivo, el usuario ha de estabilizar el dispositivo a cero grados con la ayuda de un soporte específico para teléfonos en el manillar de la bicicleta, para poder registrar la inclinación respecto al cero durante el registro de datos.

4. Con el dispositivo estabilizado a cero ya podemos iniciar el registro de ruta, el usuario puede apretar el botón central correspondiente a grabar, la aplicación empezara a tomar datos del GPS en cuanto establezca conexión (sino ha podido hacerlo aún). La aplicación empezará a registrar datos cuando los valores tomados por el sensor de ubicación sean válidos, el usuario apreciará que los campos tiempo, velocidad y distancia empiezan a mostrar valores.
5. Una vez el usuario acabe el entrenamiento a registrar, ya podrá apretar el botón izquierdo, correspondiente a el registro de la ruta, la aplicación acabará guardando los últimos valores tomados y finalizara el registro redireccionando al usuario al menú principal.

Cálculos matemáticos de la distancia entre dos puntos

Los datos que nos proporciona el sensor de ubicación de nuestro dispositivo acerca de una posición son tres: latitud, longitud y altura. App Inventor no tiene ninguna opción para determinar la distancia entre dos puntos dados por el sensor de ubicación, con lo que la aplicación ha de ser capaz de calcular la distancia aprovechando los datos ofrecidos por el sensor. El cálculo de distancia se ha hecho menospreciando la altura, esta decisión es por dos razones:

- La altura que proporcionan los sensores de ubicación de los dispositivos actuales no es precisa, por lo tanto estaríamos calculándolo con valores incorrectos.
- Aunque se tuviera una altura muy precisa, la diferencia de altitud entre dos puntos apenas afecta a la distancia entre ellos: para una pendiente del 20% , la diferencia entre la distancia horizontal y la distancia recorrida es menor del 2% (0,5% para una pendiente del 10%. Este cálculo se realiza fácilmente con el teorema de Pitágoras).

Antes de realizar ningún cálculo se ha de valorar cada cuanto la aplicación quiere obtener datos del sensor de ubicación. Ésta valoración se debe hacer planteando el problema que conlleva coger demasiados datos, (saturando la aplicación y la base de datos) y el problema de coger pocos valores (quedando el recorrido recortado en los trozos con giros).

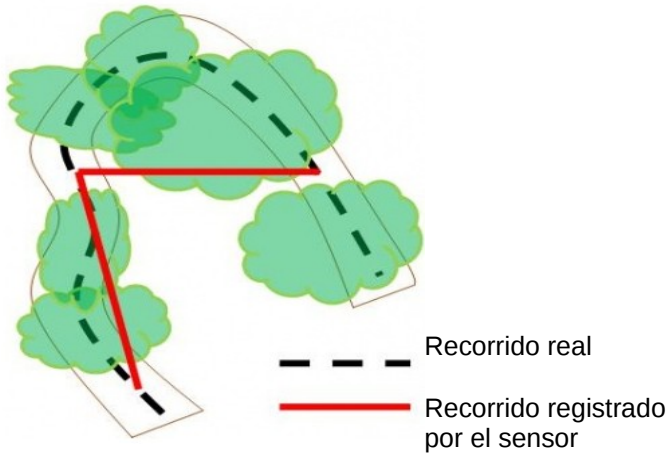


Figura 7: Diferencias entre recorrido real y estimando

Para conseguir un recorrido real no podemos coger puntos muy distantes, así que después de varias pruebas observando los puntos que obtiene el sensor, se ha valorado que un buen muestreo es coger un punto (latitud, longitud) cada tres segundos, garantizando un buen trazado sin saturar la base de datos.

Otro aspecto importante a valorar en la obtención de los datos del sensor de ubicación, es la precisión que este nos da. Los teléfonos móviles actuales disponen de sensores de ubicación no muy precisos, por eso, esta aplicación cogerá los datos de latitud y longitud después de procesar varias muestras, la aplicación coge un valor de latitud y longitud cada segundo, pero solo guardara una cada tres segundos, es decir, la aplicación calculará la media cada tres segundos de tres muestras obtenidas del sensor para garantizar una buena precisión.

Modelos matemáticos

El cálculo de la distancia entre dos puntos definidos por latitud y longitud se pueden hacer de tres maneras, estas se diferencian por el modelo matemático a escoger según la forma geométrica que valoremos que tiene la Tierra. Para decidir un modelo matemático tenemos que valorar un modelo que emule lo mejor posible a la situación real, pero valorando la complejidad de cálculo, quedando limitado por la potencia y el tiempo de cálculo.

- Esfera: considerar la tierra como una esfera es la aproximación más sencilla, pero renunciamos a la máxima precisión, pero sin duda buena por la simplicidad de la trigonometría esférica. Para el cálculo de la distancia entre dos puntos situados en la superficie de una esfera se utiliza la fórmula de Haversine, que es la más precisa frente a otras alternativas como la ley de los cosenos o el teorema de Pitágoras.
- Elipsoide: si consideramos la tierra un elipsoide, existe un algoritmo iterativo conocido como fórmulas de Vincenty que proporciona una precisión extremadamente buena. Las formulas de Vicenty al ser iterativo, puede provocar un bucle infinito en el cálculo para puntos casi opuestos.
- Geoide: los astros, como el planeta Tierra, no son esféricos sino que por efectos de la gravitación y de la fuerza centrifuga producida al rotar sobre su eje se genera el aplanamiento polar y el ensanchamiento ecuatorial. Esta valoración de la forma de la Tierra es correcta si se representa al planeta con el nivel medio de las mareas, si se considera la corteza, la Tierra no es exactamente un geoide.

Como para elegir el método de cálculo hemos de valorar la potencia y tiempo de cálculo podemos descartar el modelo de Geoide, siendo el elipsoide muy preciso y mucho menos costoso. Para acabar de decidir el modelo a elegir vemos diferentes ejemplos de cálculos con los dos tipos de modelo que no hemos descartado, esfera y elipsoide:

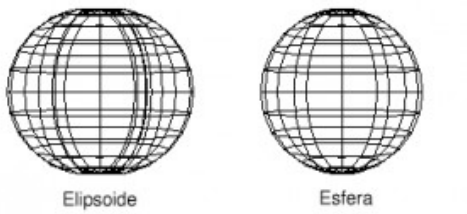


Figura 8: Modelos geométricos

Los cálculos para estos modelos se han realizado con la formula de Haversine en la esfera y las formulas de Vincenty en el elipsoide:

	Punto 1 (latitud, longitud)	Punto 2 (latitud, longitud)	Haversine	Vincenty
Puntos cercanos	43.5291675º, -5.6391020º	43.5292752º, -5.6390161º	13,83 m	13,835 m
Puntos alejados	43.5372001º, -5.6370439º	43.5402668º, -5.6475824º	915,4 m	917,392 m
Puntos muy alejados	40.452961°, -3.688333°	41.380833°, 2.122778°	499,0 km	500,15 km

Figura 9: Pantalla "Menú principal"

La aplicación calculará la distancia cada tres segundos de recorrido, por lo tanto al ciclista no le da tiempo a recorrer mucho más que unos escasos metros. En el ejemplo de los puntos cercanos vemos que perdemos tan solo cinco milímetros, esta variación es insignificante, por lo tanto, el método cogido para el cálculo de la distancia es el modelo de la Tierra como una esfera. A continuación se muestra la fórmula de Haversine y el proceso de la aplicación que realizara el cálculo de la distancia:

$$R = \textit{radio de la Tierra}$$

$$\Delta lat = lat2 - lat1$$

$$\Delta long = long2 - long1$$

$$a = \sin^2(\Delta lat/2) + \cos(lat1) \cdot \cos(lat2) \cdot \sin^2(\Delta long/2)$$

$$c = 2 \cdot \operatorname{atan2}(\sqrt{a}, \sqrt{(1-a)})$$

$$\text{Distancia (km)} = R \cdot c$$

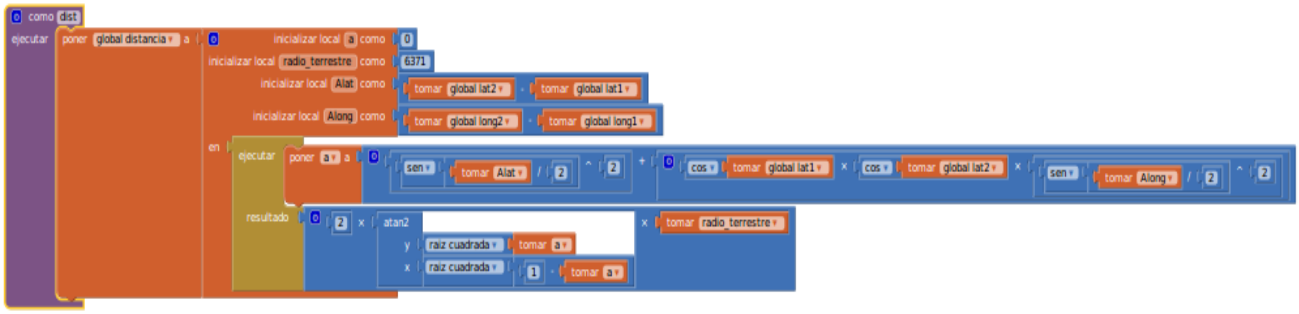


Figura 10: Bloques fórmula Haversine

4.1. Menú principal

Pantalla de inicio de aplicación, antes de acceder a ella se nos muestra un mensaje para que el usuario active el sensor GPS y el Wifi de su dispositivo. El wifi se debe conectar al de la cámara deportiva GoPro3.

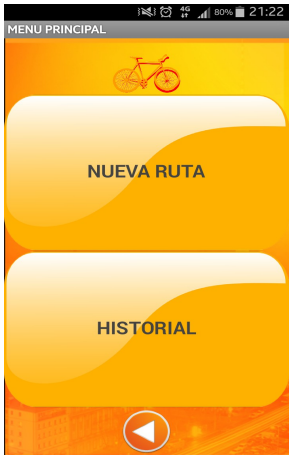


Figura 11: Pantalla "Menú principal"

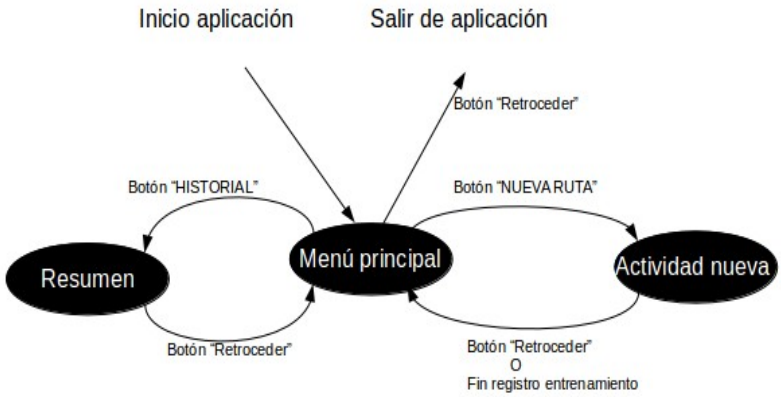


Figura 12: Diagrama de estados

En esta primera pantalla tenemos tres opciones:

- Botón "NUEVA RUTA", nos abrirá la pantalla correspondiente al registro de un nuevo entrenamiento "Actividad nueva".

Figura 13: Bloques botón "nueva_ruta"

- Botón "HISTORIAL", este botón es en verdad un listado, al seleccionarlo abrirá una lista con los entrenamientos guardados en la base de datos "TinyBD1". El usuario selecciona una de las rutas guardas y a continuación se abrirá la pantalla "RESUMEN" correspondiente a la ruta seleccionada.

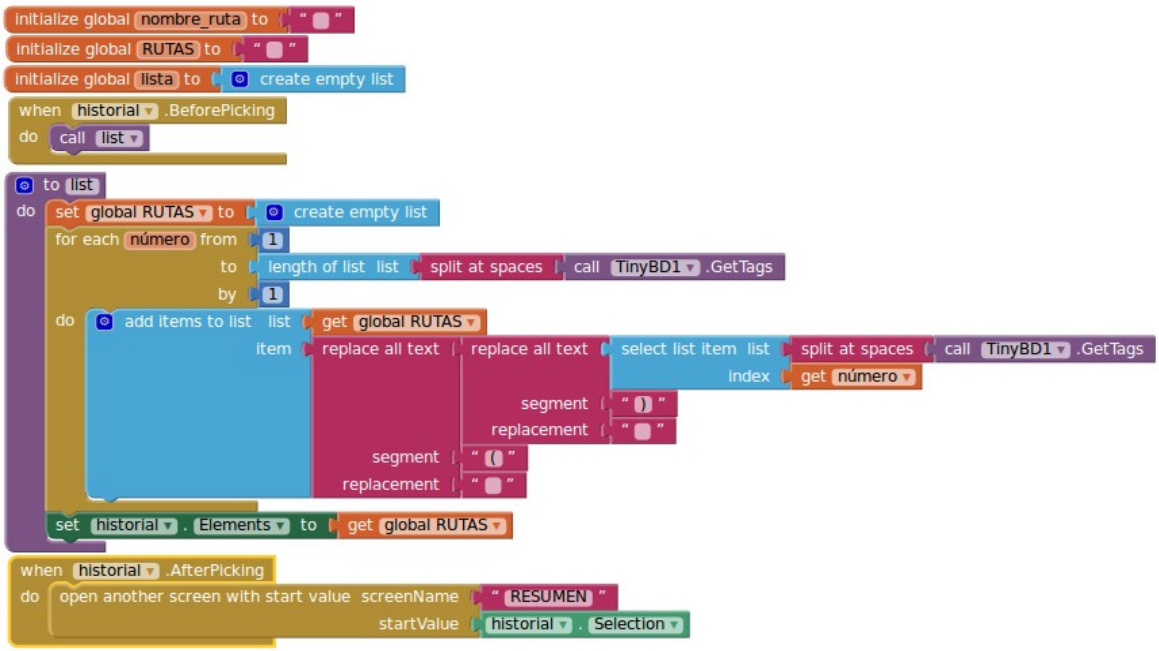


Figura 14: Bloques botón"HISTORIAL"

- Botón “salir”, este botón que se encuentra en el inferior de la pantalla nos cerrara la aplicación.

Figura 15: Bloque botón “salir”

4.2. Actividad nueva

Esta pantalla es la más compleja de la aplicación, tiene como finalidad guardar una actividad nueva sincronizándose con la cámara deportiva, genera el fichero “nombre_ruta.txt” con toda la información de la ruta y guardar un nuevo registro en la base de datos de la aplicación.

Figura 16: Pantalla “Menú principal”

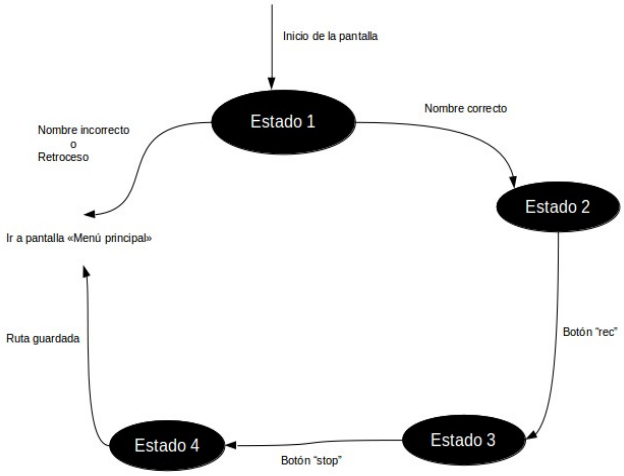


Figura 17: Diagrama de estados

- Estado 1, este estado enciende la cámara introduciendo una la url “http://<ip>/bacpac/PW?t=<password>&p=%01” en un navegador web oculto, habilita los sensores de ubicación y de orientación y el temporizador “grados”. También nos abre un diálogo donde el usuario ha de introducir un nombre para la ruta, este campo no puede estar vacío.

  El sensor de Ubicación esta configurado previamente para coger una muestra por segundo. El temporizador “grados” nos muestra por pantalla el valor que recoge el sensor de orientación cada 200ms. Para poder mostrar el resultado acorde a la posición del dispositivo se invierte el valor tomado.

- Estado 2, si el usuario ha dado un nombre correcto la aplicación crea el archivo “nombre_ruta.txt” en el directorio “/sdcar/” de nuestro dispositivo y guarda un primera linea con la hora del sistema actual con el formato “<hora_del_sistema”, centinela “<” nos indica el inicio del fichero. Ejemplo:

<1429710415988

Figura 18: Archivo "nombre_ruta.txt"

Dentro de este estado se envía otra url con el formato “http://<ip>/camera/CM?t=<password>&p=%00” para poner la GoPro en modo cámara de vídeo.

- Estado 3, cuando el usuario tiene el dispositivo totalmente vertical ha de seleccionar el botón “rec” y el registro de la ruta empieza su proceso. La aplicación espera a establecer conexión del sensor de Ubicación para empezar el registro.

  Una vez establece conexión GPS lee una lectura de de longitud y una de latitud por segundo y durante seis segundos la aplicación coge valores para sus variables de cálculo antes de empezar. Una vez pasan esos seis segundos las variables ya tienen valores de la localización actual y la aplicación inicia el proceso de registro.

  Cada tres segundo controlado por un temporizador hace la media entre el cúmulo de cada una de las variables de ubicación, este calculo se realiza para garantizar una precisión aceptable del sensor. Los sensores de Ubicación de de los dispositivos tienen una precisión aceptable, pero no suficiente, por eso se procesa esta información para garantizar una precisión mucho mejor. Este temporizador también se encarga de calcular los grados actuales, el sensor de ubicación recoge muestras cada 200ms, así que se filtra para conseguir un valor mucho más preciso de la inclinación, es decir, el valor de inclinación es el resultado de 15 muestras, de esta manera absorbemos las vibración provocadas por el movimiento.

  El temporizador también se encarga del cálculo de la distancia cada tres segundos y de mostrar en el formato correcto los datos por pantalla a la vez de guardarlos en el archivo de la ruta “/sdcar/”nombre_ruta.txt” y ir construyendo una estructura con los datos necesarios para la pantalla “RESUMEN” que guardará en la base de datos. Los datos guardados en el fichero son tres; tiempo, latitud, longitud y la inclinación del dispositivo. Todos los datos guardados forman una trama por linea cada tres segundos, separados por una almohadilla.

<1429710415988
2#41.54041#1.899#3
5#41.54041#1.899#6
8#41.54041#1.899#11

Figura 19: Archivo "nombre_ruta.txt"

- Estado 4, este estado es el último de esta pantalla, se inicia cuando el usuario selecciona el botón de parar el registro de ruta, interpretado por el la imagen de “stop”. La aplicación deshabilita todos los sensores y procesos, guarda en la base de datos de la aplicación todos los datos necesarios y guarda una última línea en el archivo “nombre_ruta.txt” para guardar los últimos datos y indicar el final de ruta. La última linea del archivo contiene un centinela de final de ruta “>”, seguido por el tiempo total en segundos y la distancia total en kilómetros.

```
<1429710415988
2#41.54041#1.899#3
5#41.54041#1.899#6
8#41.54041#1.899#11
....
>2006#6.07862
```

Figura 20: Archivo "nombre_ruta.txt"

Dentro de este estado se envía otra url al visor web oculto con el formato “http://<ip>/bacpac/SH?t=<password>&p=%00” para poner finalizar la captura de vídeo de la cámara. La aplicación una vez finalizada todas estas tareas abre a la pantalla de inicio “Menú principal”.

4.3. RESUMEN

Esta pantalla es la encargada de mostrar al usuario los datos de la ruta seleccionada. Cuando en la pantalla “MENÚ PRINCIPAL” seleccionamos el botón del historial se nos abre una lista con todas las rutas contenidas en la base de datos, cuando el usuario selecciona una ruta se pasa a la pantalla “RESUMEN” como variable y esta la utiliza para extraer la información.

Figura 21: Pantalla “Menú principal”

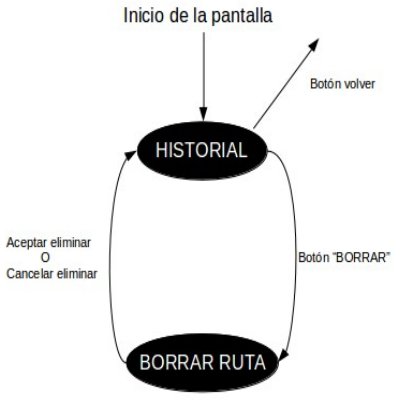


Figura 22: Diagrama de estados

En esta pantalla se nos muestran dos botones en la parte superior, el de volver al menú principal y el representado con una calavera correspondiente a borrar. También nos muestra tres valores de la ruta, distancia y tiempo extraídos de la base de datos y la media de velocidad, calculada con los datos de la base de datos. Por último la pantalla muestra la ruta dibujada sobre google maps, para poder realizar esta representación el dispositivo necesitara tener acceso a Internet porque se basa en un script utilizando la api de google maps. El código html donde se incluye el script de la api de google es como el siguiente ejemplo:

```
<!DOCTYPE html>
<html>
  <head>
    <title>Simple Map</title>
    <meta name="viewport" content="initial-scale=1.0, user-scalable=no">
    <meta charset="utf-8">
    <style>
      html, body, #map-canvas {
      height: 100%;
      width: 100%
      margin: 0px;
      padding: 0px
      }
    </style>
    <script src="https://maps.googleapis.com/maps/api/js?v=3.exp&signed_in=true"></script>
    <script>
      var map;
      function initialize() {
      //Configuramos las opciones indicando Zoom, punto centro  y tipo de mapa
      var mapOptions = {
      zoom: 17,
      center: new google.maps.LatLng(41.5413906,1.8980966),
      mapTypeId: google.maps.MapTypeId.SATELLITE
      //mapTypeId: google.maps.MapTypeId.SATELLITE define que nuestro tipo de mapa ROADMAP,SATELLITE, HYBRID o TERRAIN
      };
      map = new google.maps.Map(document.getElementById('map-canvas'),
      mapOptions);
      //Definimos los puntos de la ruta a representar
      var flightPlanCoordinates = [
      new google.maps.LatLng(41.5413906, 1.8980966),
      new google.maps.LatLng(41.5415873, 1.8971685),
      new google.maps.LatLng(41.5408927,1.8969164),
      new google.maps.LatLng(41.5410814,1.8959776)
      ];
      //Configuramos el formato a mostrar; unión entre puntos, color, etc.
      var flightPath = new google.maps.Polyline({
      path: flightPlanCoordinates,
      geodesic: true,
      strokeColor: '#FF0000',
      strokeOpacity: 1.0,
      strokeWeight: 2
      });
      flightPath.setMap(map);
      //Mostramos el marcador en el punto final de la ruta
      var marker = new google.maps.Marker({
      position:new google.maps.LatLng(41.5410814,1.8959776),
      map: map,
      title:"Inici"});
      }
      google.maps.event.addDomListener(window, 'load', initialize);
    </script>
  </head>
  <body>
    <div id="map-canvas"></div>
  </body>
</html>
```

Figura 23: Ejemplo ruta con Google maps api

La pantalla construye un string con unas variables pre definidas que encapsulan parte de este código y con datos extraídos de la base de datos, entre todos los datos construye un único archivo con la estructura anterior pero con los datos de la ruta en particular y guarda este archivo en la carpeta oculta de la aplicación (sobrescribiendo el anterior creado por la visualización de otra ruta o generando de nuevo en caso de ser la primera ruta visualizada). El archivo creado y guardado se carga en un visualizador web por el cual el usuario podrá observar el mapa de la ruta y interaccionar con el, como el Google Maps normal.

4.5. Componentes

Para la utilización de la aplicación Android SimBike son necesarios los siguientes componentes:

Dispositivo Android

Para la utilización de la aplicación es necesario un dispositivo con sistema Operativo Android. App Inventor compila la aplicación para todas las versiones de Android, pero la lección esta limitada por dispositivos con sensor de ubicación, conexión Wifi y sensor de angulo.

Figura 24: Soporte teléfono manillar

También es necesario un soporte para este dispositivo, con la finalidad de fijarlo en el la bicicleta con la que se va a realizar la grabación del entrenamiento. El soporte usado es el siguiente:

Cámara de deporte

La cámara de deporte utilizada para registrar el vídeo de la marca Gopro, modelo 3 White. Para poder realizar la sincronización con la aplicación se ha usado la conectividad wifi que incorpora este modelo de cámara. La cámara permite una gran cantidad de ajustes y acciones vía wifi, para realizar-los se ha de introducir una URL con protocolo http en un navegador web. A continuación se exponen las URL que necesitaremos para nuestra aplicación:

Figura 25: GoPro 3 white

- Apagar la cámara : http://<ip>/bacpac/PW?t=<password>&p=%00
- Encender la cámara : http://<ip>/bacpac/PW?t=<password>&p=%01
- Iniciar la grabación : http://<ip>/bacpac/SH?t=<password>&p=%01
- Parar la grabación : http://<ip>/bacpac/SH?t=<password>&p=%00
- Modo cámara vídeo : http://<ip>/camera/CM?t=<password>&p=%0
- Resolución de vídeo, la resolución escogida es esta por su buena calidad y por tener un tamaño de archivo no moderado: WVGA-60 : http://<ip>/camera/VR?t=<password>&p=%00

Figura 26: Soporte en manillar

Estas URL solo serán recibidas y interpretadas por el dispositivo con el wifi encendido, este se activa manualmente con el botón que incorpora en el lateral. También es necesario un soporte para esta cámara, se han utilizado dos sistemas de sujeción:

- Soporte en manillar, este tiene el inconveniente de estar más expuesto al movimiento de conducción.
- Arnés de sujeción, aporta más realismo al vídeo gracias a tener una perspectiva desde el pecho del ciclista y ver el manillar.

Figura 27: Arnés

5. Emulador de entrenamiento

El emulador de entrenamiento es la segunda gran parte de este proyecto, es el encargado de emular las rutas con los datos generados anteriormente con la aplicación Android. Este emulador tiene las funciones siguientes:

- Interaccionar con el usuario para recibir toda la información de los entrenamientos. Una vez tenga un entrenamiento registrado el usuario posee un fichero en el dispositivo realizado con toda la información del entrenamiento y un vídeo de éste en la cámara, ahora el usuario solo tiene que enviar estos dos archivos al emulador. Para realizar esta labor el emulador proporciona al usuario una interfaz gráfica para poder recibir todos los datos, por lo tanto solo tiene que conectarse con un dispositivo que contenga los dos archivos a la red Wifi creada por el emulador y seguir intuitivamente la aplicación web.
- Guarda la información y el vídeo de los entrenamientos en una base de datos para ofrecer al usuario un historial de todos los entrenamientos guardados y poder emularlos siempre que el usuario quiera realizarlos.
- Ofrecer una interfaz gráfica intuitiva para emular los entrenamientos guardados. Para emular un entrenamiento el usuario solo necesita un dispositivo que contenga un navegador web y tenga conexión Wifi.

5.1. Interfaz Gráfica

El emulador de entrenamientos se ha diseñado como una aplicación web, la cual el usuario tiene a nivel local gracias a una red Wifi que crea el emulador. Esta aplicación web esta diseñada para ofrecer una

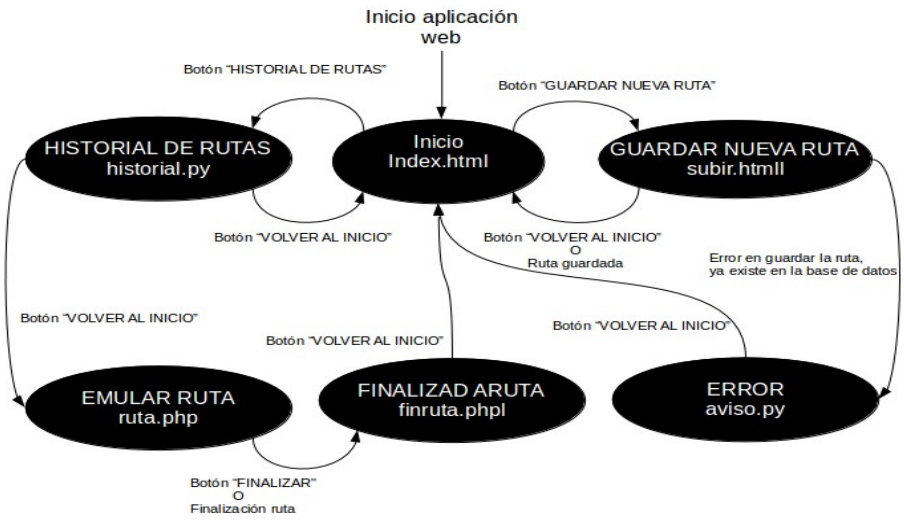


Figura 28: Pantallas aplicación web

interfaz gráfica al usuario con una fácil comprensión y un uso muy simple. La aplicación se compone de cuatro pantallas, la primera es la de inicio con dos botones que abren las otras dos, una para guardar las rutas y otra para consultar el historial y redirigir a la cuarta pantalla de emulación.

Pantalla de inicio

Esta pantalla es la de inicio de la aplicación, es de extrema simplicidad y solo tiene dos botones para la interacción con el usuario. Para iniciar la aplicación el usuario tiene que conectarse con el emulador mediante al Wifi creado por este (el SSID de la red Wifi es “SIMBIKE” y la contraseña “emulador15”). Una vez el usuario tiene acceso a la red “SIMBIKE” tiene que introducir en un navegador web la IP:192.168.100.1 correspondiente al del dispositivo emulador, el resultado es la pagina de inicio “index.html”:

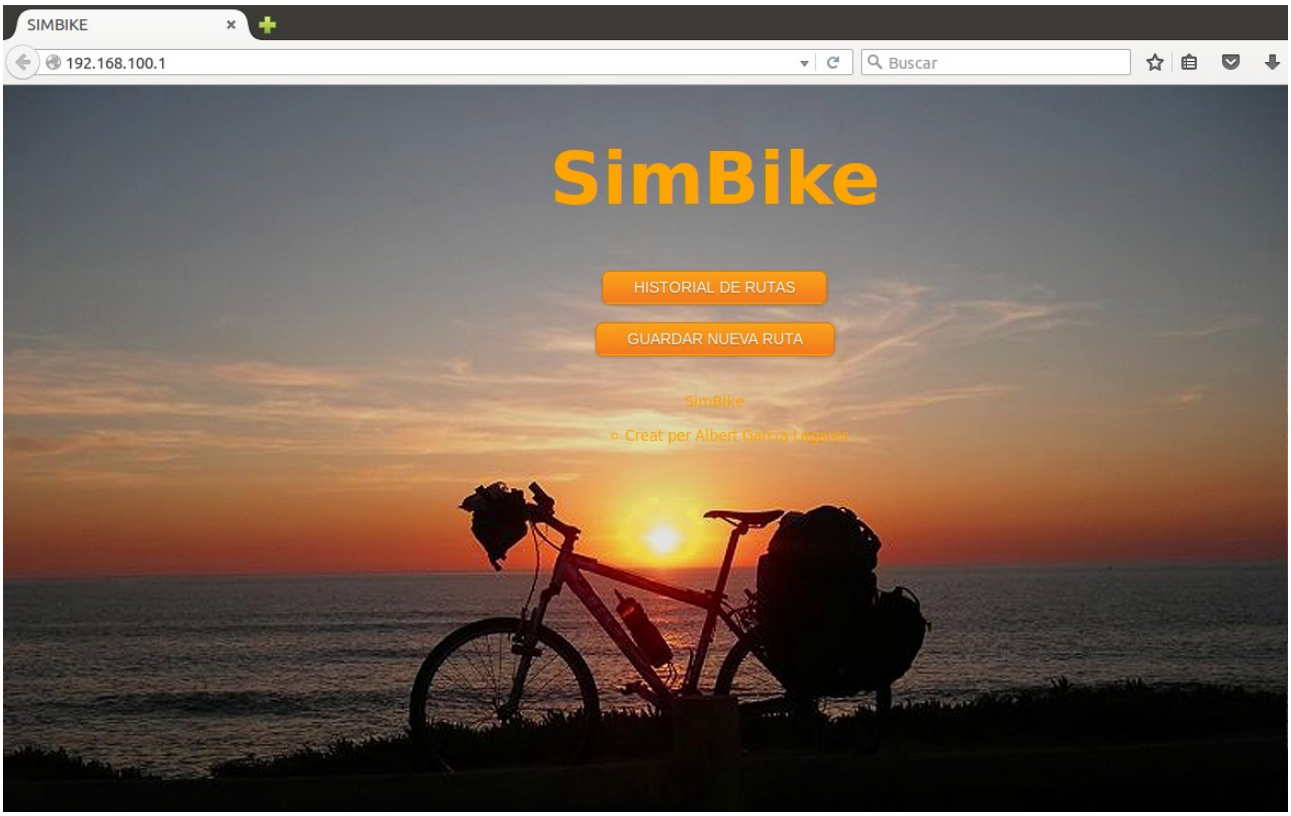


Figura 29: index.html

Los botones que muestra la pantalla de inicio son “HISTORIAL DE RUTAS” que cargará la pantalla “historial.py”, para visualizar el historial de entrenamientos guardados y “GUARDAR RUTAS” que muestra a “subir.html” para guardar en la base de datos una nueva ruta.

Pantalla historial de rutas

En esta pantalla se muestran las rutas guardadas en una tabla donde podemos ver los datos de ésta. En ella el usuario puede gestionar su base de datos eliminado todas las rutas, eliminando una de ellas o seleccionando una ruta para emularla. También se ofrece un botón para volver a la página de inicio.

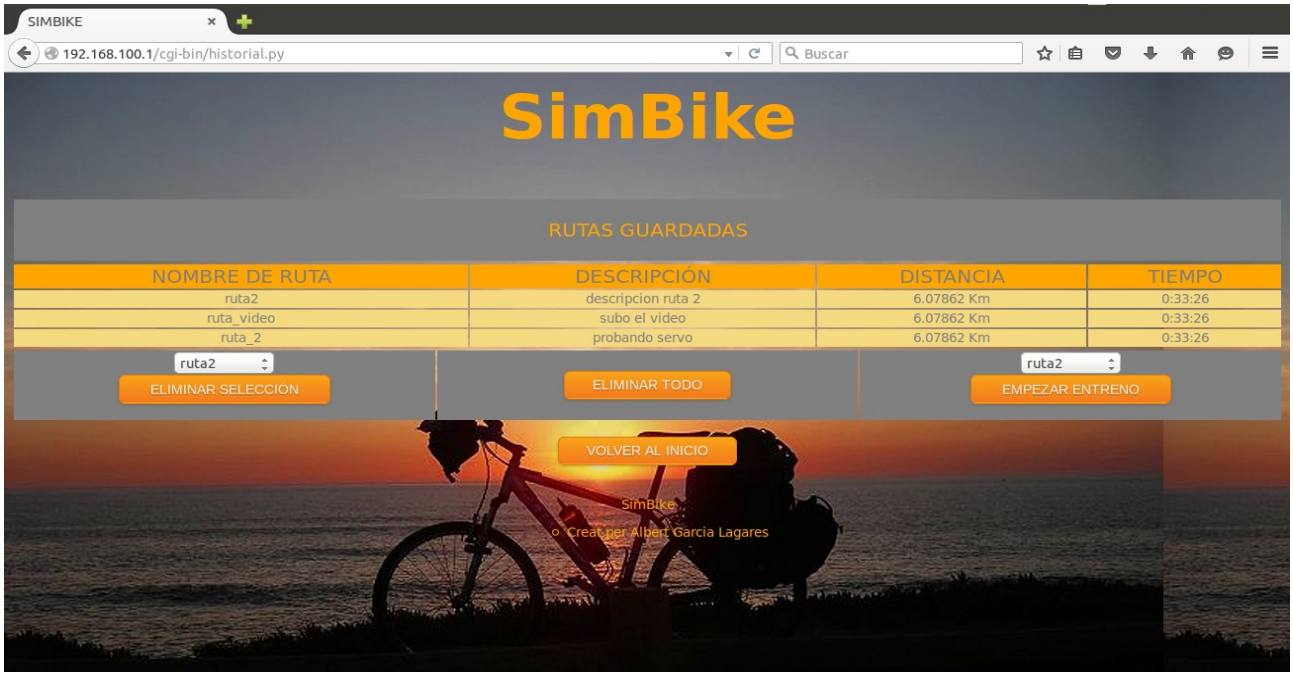


Figura 30: historial.py

Pantalla de emulación

Esta pantalla nos muestra cómo se divide en tres columnas: la primera son los datos de la ruta a emular, la segunda el tiempo que esta transcurriendo en la emulación y la tercera el vídeo de la ruta que estamos emulando. El usuario tiene a su disposición un botón para finalizar la ruta cuando lo desee si no quiere acabar.

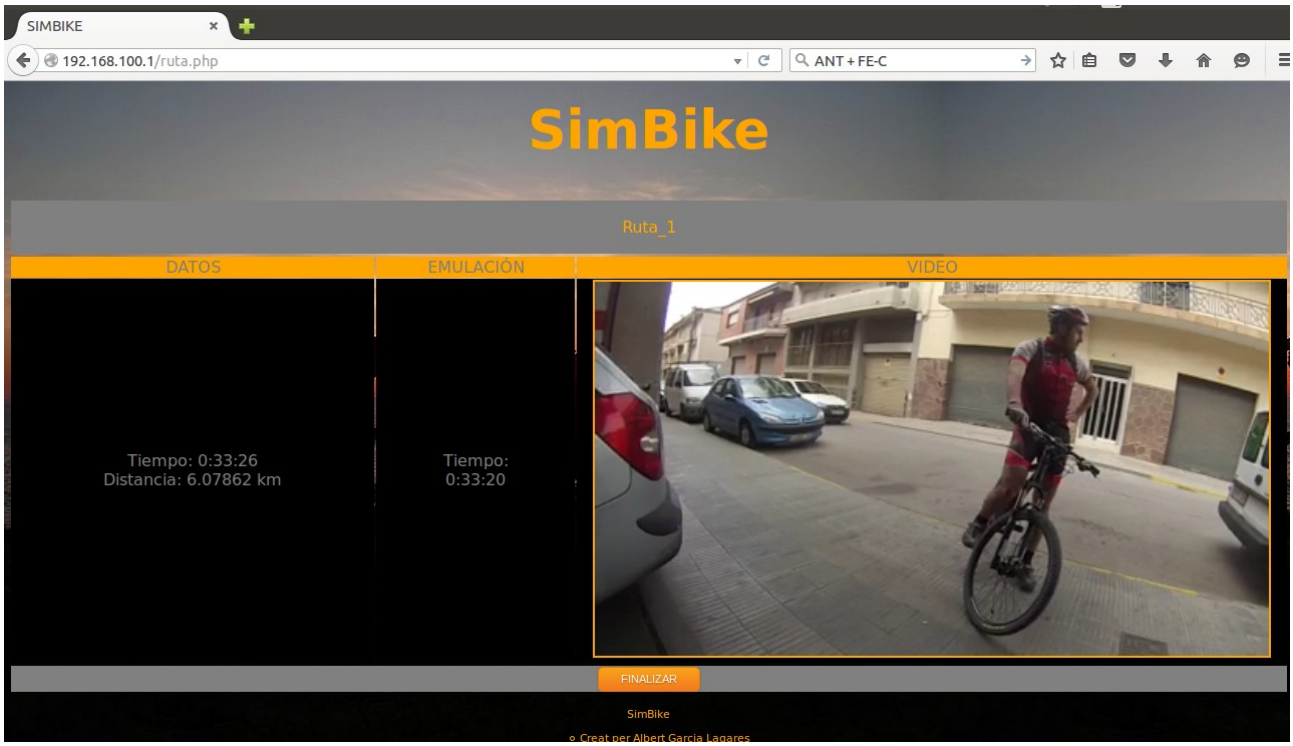


Figura 31: ruta.php

5.2. Estructura del emulador

El emulador contiene gran cantidad de módulos y ficheros dada su complejidad. A continuación se va detallar todos los elementos que intervienen en esta aplicación web y la relación entre ellos:

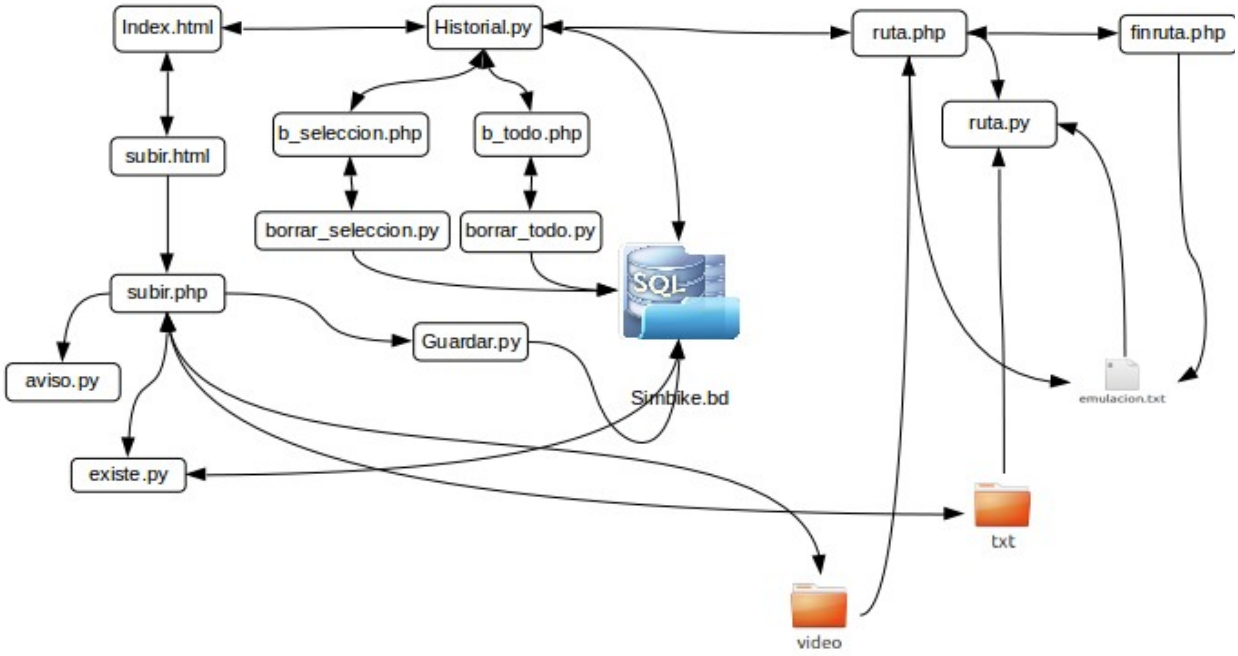


Figura 32: Estructura simulador web

- Index.html : página de inicio de la aplicación web, incluye dos botones para navegar por la aplicación, uno para ir a la página de guardar nueva ruta y otro para ir al historial de rutas guardadas. El código usado para ésta ha sido html.
- Subir.html : en esta página tenemos dos campos a rellenar (nombre y descripción de la ruta) y fichero a introducir (fichero de datos de la ruta y vídeo de la ruta). Esta página esta elaborada con html usando un formulario, este formulario obliga al usuario a introducir todos los campos menos la descripción y envía todo la información con el método POST.
- Subir.php : este php se encarga de guardar los dos archivos anteriormente introducidos dentro de los directorios correspondientes. Los archivos guardados los nombra con el nombre de ruta y la extensión correspondiente (txt y mp4). Antes de proceder a guardar la información comprueba si existen con el valor devuelto por el modulo “existe.py”, el cual devuelve “si” o “no” según a la existencia en la base de datos del nombre de ruta pasado como argumento.
- Existe.py : este se encarga de consultar en la base de datos y retornar si existe o no el nombre de esta ruta.

- Aviso.py : página que muestra que la ruta con nombre introducido ya existe. “subir.php” le pasa con el método POST el nombre de la ruta.
- Guardar.py : se encarga de guardar en la base de datos el nombre de la ruta como identificador y la descripción de esta.
- Historial.py : nos muestra una tabla con el contenido de la base de datos “simbike.bd” y las informaciones de esta (kilómetros y tiempo del recorrido), estas informaciones las extrae del fichero con formato de texto.
- b_selección.php : la única finalidad de este es ejecutar el fichero python que eliminará la selección hecha en la página historial. El nombre de la ruta se le envía desde “historial.py” con el método POST.
- b_todo.php : como el anterior se encarga de ejecutar el python que eliminará toda la base de datos.
- borrar_seleccion.py : se encarga de borrar de la base de datos la ruta seleccionada que se le ha pasado como argumento. También borra el fichero de texto de la ruta y el vídeo de esta.
- b_todo.py : se encarga de borrar de la base de datos todas las rutas, así como los ficheros de texto de las rutas y los vídeos.
- Ruta.php : esta página emula la ruta pasada con el método POST desde la página del historial. Esta ejecuta paralelamente el programa python (“ruta.py”) encargado del control del servo. En esta se muestran los datos de la ruta y el vídeo, la reproducción se ha creado con el reproductor de vídeo html5 y controlado por javascript.
- Ruta.py : este programa se encarga de interpretar el fichero correspondiente que se la ha pasado como argumento y controlar el servo para seleccionar el nivel de esfuerzo del rodillo de entrenamiento.
- Finruta.php : este código tiene la función de mostrar al usuario la finalización de la ruta y escribir en el fichero “emulacion.txt” con la finalidad de comunicarse con el programa python en ejecución.
- Emulacion.txt : este fichero alberga un único carácter que puede ser “0” o “1”. El archivo “ruta.php” modifica a “1” el contenido del fichero cuando se inicie la emulación de una nueva ruta y “finruta.php” sobrescribirá el contenido dejando “0” cuando se finalice la emulación. La información del archivo es consultado en todo momento por el programa python “ruta.py” para saber si ha de acabar su proceso de emulación.
- Simbike.bd : es la base de datos de la aplicación, tiene la función de guardar el nombre de ruta como clave primaria y la descripción de las rutas como valor.
- Directorios: los directorios “txt” y “videos” albergan los fichero de información de las rutas y los vídeos de estas.

5.3. Aplicación web

La aplicación web diseñada se ha elaborado para facilitar el aprendizaje al usuario, siendo esta de muy fácil comprensión y utilización. La ventaja de ser una aplicación web es la gran compatibilidad que disponemos para poder emular el entrenamiento en muchos dispositivos. Cualquier dispositivo con navegador web y WiFI es un buen candidato para poder emular un entrenamiento o para subir los datos al emulador (servidor).

Otra gran ventaja de crear el emulador sobre el formato de aplicación web es la cantidad de recursos que podemos utilizar a la vez, prueba de esto es este proyecto en el que se han integrado cinco tipos de lenguajes diferentes, aprovechando los puntos fuertes de cada uno de ellos. Los lenguajes de programación usados son:

- PHP
- Python
- HTML
- Javascript
- SQL

5.3.1. PHP

PHP es un lenguaje de código abierto muy popular especialmente adecuado para el desarrollo web. Este lenguaje se ejecuta en el servidor, el cliente recibirá el resultado de ejecutar el script php sin saber que se esta ejecutando en el servidor. En el proyecto se recurre a el para realizar diferentes funciones, gracias a ejecutarse en el servidor nos da la facilidad de interaccionar con los programas y ficheros que necesitemos, se podría definir la función de este lenguaje en el proyecto como enlace entre la interfaz gráfica y la ejecución del programa.

Usos de php en la aplicación web

Los usos que se le han dado a este lenguaje son varias y muy importantes para el buen funcionamiento de la aplicación web:

- Subida de archivos al servidor, cuando el usuario en la pantalla de guardar nueva ruta selecciona los dos archivos éstos se pasan con el método HTTP POST a “subi.php”, en este archivo se usan los métodos :
    - bool is_uploaded_file (string $filename) . Devuelve TRUE si el archivo nombrado por filename fue subido mediante HTTP POST.

        - bool move_uploaded_file (string $filename , string $destination). Esta función asegura que el archivo designado por filename es un archivo subido válido (subido mediante el mecanismo de subida HTTP POST de PHP). Si el archivo es válido, será movido al nombre de archivo dado por destination.

- Comprobación de permisos, cuando subimos un archivo al servidor comprobamos si el directorio tiene permisos con:

    - bool is_writable (string $filename), devuelvo TRUE si tenemos permisos de escritura en el directorio donde queremos guardar el archivo definido por la variable filename.

- Ejecutar scripts python, durante el uso de la aplicación web se requiere información o ejecución de algunos programas python. Para realizar la ejecución de estos programas python se ha recurrido a la función:

    - string exec (string `$command` [, array `&$output` [, int `&$return_var`]]). Éste ejecutara un comando externo con la facilidad de adjuntar valores de salida para poder pasar argumentos necesarios y después de la ejecución recoge el valor devuelto por si necesitamos información del proceso ejecutado.

      El uso de exec se ha realizado en los módulos; subi.php, b_todo.php, b_seleccion.php y ruta.php

- Imprimir código html con fragmentos de php, una de las particularidades del lenguaje php es la facilidad de incrustar trozos de código en los fichero html, el resultado de esto es que el usuario observa el resultado de esta ejecución sin saber que php se esta ejecutando en el lado del servidor.

Configuraciones importantes de php

Las configuraciones necesarias para la ejecución de nuestro código PHP son:

- En la subida de archivos al servidor con los métodos anteriormente citados, se tiene que tener en cuenta los permisos del directorio, estos tienen que tener permisos de escritura.

- Para poder realizar la subida de archivos se han tenido que modificar el archivo de configuración de php “/etc/php5/apache2/php.ini”. Esta configuración es a causa que por defecto el tamaño de archivo máximo que acepta para su subida es de dos megas y nuestros vídeos ocupan alrededor de 180 megas. Para poder realizar la subida de archivos grandes se han tenido que cambiar las líneas correspondientes a la subida de archivos por php y la de máximo tamaño del método POST, de 2 megas a 190 megas. Este cambio nos permitirá subir archivos de vídeo de un poco más de media hora, si se quisiese aumentar solo se tendría que dar más margen.

- Para poder utilizar el comando *exec* de PHP tenemos que otorgar permisos.

```
; Maximum allowed size for uploaded files.
; http://php.net/upload-max-filesize
upload_max_filesize = 2M

; Maximum number of files that can be uploaded via a single request
max_file_uploads = 20
```

```
; Maximum size of POST data that PHP will accept.
; Its value may be 0 to disable the limit. It is ignored if POST data reading
; is disabled through enable_post_data_reading.
; http://php.net/post-max-size
post_max_size = 190M
```

Figura 33:php.ini

5.3.2. Python

El lenguaje python dentro del proyecto tiene diferentes y variadas funciones. Una de la principales razones del uso de este lenguaje es por la facilidad de interacción con la base de datos, python es el único lenguaje en toda la aplicación web que interacciona con la base de datos SQL. También se ha usado para crear scripts python para crear una interfaz gráfica dinámica para el usuario. A continuación se detalla el uso de python en la aplicación:

- Scripts python, para la pantalla historial se ha usado el lenguaje python para la consulta a la base de datos y imprimir todo el html necesario.
- Control de puertos GPIO de la Raspberry, para el control de del servomotor se ha usado los puertos GPIO y para controlarlos se ha realizado con el módulo python GPIO. Las funciones usadas de este módulo son:
    - GPIO.setmode(GPIO.BCM). Configuramos el modo en BCM para poder referirnos a los pines en modo BCM, también podemos hacerlo en modo BOARD, teniendo en cuenta que la numeración es totalmente distinta.
    - GPIO.setup(9,GPIO.OUT) . Configuramos como salida el pin nueve en el cual esta el servomotor conectado.
    - p = GPIO.PWM(9,50). Configuramos la salida del pin nueve como un señal PWM de 50 pulsos por segundo, tal y como el servo necesita.
    - p.start(7.5). Con start iniciamos los ciclos PWM de salida con un 7'5% del período en valor alto.
    - p.ChangeDutyCycle(4.5). Esta función nos permite cambiar el porcentaje del tiempo de ciclo de salida.

- Gestión de la base de datos. La facilidad de uso de la base de datos SQL desde python ha ayudado mucho a elaborar un código muy simple. Necesario en los archivos python de guardar y borrar rutas.
- Gestión de archivos. Este es otro de los puntos fuertes de python, la facilidad de gestión de los archivos que tenemos en el servidor, esencial para la emulación desde ruta.py.

Configuraciones importantes de Python

La configuración importante para nuestro código python en la aplicación es la configuración de apache para la ejecución de scripts python de nuestro servidor, modificando el archivo “/etc/apache2/sites-enabled/000-default” añadiendo la línea “AddHandler cgi-script.py”, como se muestra a continuación:

```
ScriptAlias /cgi-bin/ /usr/lib/cgi-bin/
<Directory "/usr/lib/cgi-bin">
        AllowOverride None
        Options +ExecCGI -MultiViews +SymLinksIfOwnerMatch
        Order allow,deny
        Allow from all
        AddHandler cgi-script .py
</Directory>
```

Figura 34:000-default

Después de esta modificación tendremos que reiniciar el servidor y el servidor ya nos mostrara el resultado de ejecutar los escripts python contenidos en la carpeta cgi-bin. Por defecto los scripts-cgi se ubican en el directorio “/usr/lib/cgi-bin ”.

5.3.3. Html5

Este lenguaje es esencial para la aplicación, se ha usado para crear la interfaz gráfica. Los lenguajes PHP y Python en esta aplicación web cuando tienen que mostrar información lo hacen imprimiendo código html, por lo tanto es el lenguaje usado para toda la interfaz gráfica. Dentro de este lenguaje se han usado los recursos de formato ofrecidos por el fichero formats.css, recursos de formularios y el método HTTP POST, pero sin duda por la importancia en esta aplicación web el recurso más importante utilizado el la etiqueta de vídeo, esta etiqueta no es asta esta revisión 5 que se ha incorporado.

La etiqueta de vídeo se suele presentar como alternativa a Flash para el contenido multimedia, pero tiene más aplicaciones. Sus posibilidades y su compatibilidad con diferentes navegadores han aumentado a una increíble velocidad, haciendo al vídeo html muy utilizado. La ventaja principal de la etiqueta de vídeo es la posibilidad de integración natural con las demás capas de la pila de desarrollo web, como CSS y JavaScript, así como con las otras etiquetas HTML. La etiqueta html de vídeo usada en ruta.php para la visualización del vídeo contiene algunos parámetros de configuración, tamaño, borde y el más importante es el autoplay (para la reproducción automático de nuestro vídeo), también se han deshabilitado los controles de vídeo y conseguir así que el usuario no pueda controlar el vídeo, garantizando que sea el programa quien controla y sincroniza el vídeo con la emulación física del entrenamiento. La etiqueta de vídeo es:

<video id="Video1" style="border: 3px solid orange;" height=95% width=95% title="video element" autoplay></video>

5.3.4. Javascript

El uso del JavaScript en este proyecto no es muy extensa, aunque si necesaria.

- “historial.py” se utiliza JavaScript para dar formato de “horas:minutos:segundos” al tiempo registrado en segundos extraído del fichero nombre_ruta.txt de las rutas guardas y el uso más importante es en el reproductor de vídeo html

```
<script language="javascript">
        var segundos = <?php echo $t; ?>;//Segundos de la cuenta atrás
        function tiempo(){
                var t = setTimeout("tiempo()",1000);
                var horas1 = Math.floor( segundos / 3600);
                var minutos1 = Math.floor( segundos / 60);
                var segundos1 = segundos % 60;
                minutos1 = minutos1 < 10 ? "0" + minutos1:minutos1;
                segundos1 = segundos1 < 10 ? "0" + segundos1:segundos1;
                var reloj = horas1 + ":" + minutos1 + ":" + segundos1;
                segundos--;
                document.getElementById("reloj2").innerHTML = reloj;
            if (segundos==0){
              window.location.href='finruta.html';  //Págiana a la que redireccionará a X segundos
        clearTimeout(t);
        }
        }
        tiempo()
</script>
```

Figura 35:000-default

- “ruta.php”, en el reproductor de vídeo html se usa JavaScript para definir la ruta del archivo MP4 a reproducir.

```
<script type="text/javascript">
    function init() {
      var video = document.getElementById("Video1");
      video.src = "<?php echo $ruta_video; ?>";
      video.load();
    }
  </script>
```

Figura 36:000-default

5.3.5. SQLite

La aplicación web no tiene la necesidad de una base de datos, los datos guardados son muy pocos pero la facilidad de utilización de la base de datos hace atractiva la incorporación de esta en el proyecto, dándole la posibilidad de futuras modificaciones con el recurso ya creado de esta base de datos. La base de datos de la aplicación se ha definido con el lenguaje SQLite por sus numerosas ventajas:

- Tamaño: SQLite tiene una pequeña memoria y una única biblioteca es necesaria para acceder a bases de datos, lo que lo hace ideal para aplicaciones de bases de datos incorporadas.
- Rendimiento de base de datos: SQLite realiza operaciones de manera eficiente y es más rápido que MySQL y PostgreSQL.
- Portabilidad: se ejecuta en muchas plataformas y sus bases de datos pueden ser fácilmente portadas sin ninguna configuración o administración.
- Interfaces: cuenta con diferentes interfaces del API, las cuales permiten trabajar con C++, PHP, Perl, Python, Ruby, Tcl, groovy, etc. En este caso se trabajará con Python.
- Costo: SQLite es de dominio público, y por tanto, es libre de utilizar para cualquier propósito sin costo.

Nuestro fichero de la base de datos “db.sql” contiene la definición de una única tabla con dos valores y el archivo creado es nombrado “como simbike.bd”, a continuación se muestra la table que incluye.:

```
CREATE TABLE RUTAS (
    id_ruta VARCHAR PRIMARY KEY NOT NULL,
    descripcion VARCHAR NULL
);
```

Figura 37:Definición base se datos con sqlite

5.4. Servidor web

El servidor web se ha instalado en la Raspberry pi, con el sistema operativo Raspbian. El servidor es Apache2, este servidor es web HTTP de código abierto compatible con plataforma UNIX. Las ventajas de este servidor es la alta configuración que dispone y que permite ser extendido con módulos externos, esta característica es muy importante para el proyecto porque se necesitan los módulos:

- mod_python, para crear páginas dinámicas con python.
- mod_php, para crear páginas dinámicas con php.

5.5. Punto de acceso Wifi

El emulador esta configurado como punto de acceso wifi, esta característica nos permite conectarnos a la red wifi que genera el emulador y poder consultar la aplicación web. La elección de esta configuración es por la necesidad de comunicación con el servidor web para acceder a la aplicación web, pero sin la necesidad de tener un router para enlazar la comunicación entre emulador y dispositivo, ni de tener que subir nuestra web a ningún servidor externo. De esta manera no tendremos ni que vincular el emulador a una red local ni a internet. La configuración como punto de acceso Wifi se ha realizado con los programas Hostaps y DHCP.

Para poder realizarEn primer lugar instalaremos los programas que nos permitirán crear el punto de acceso Wifi, estos son dos:

- Hostapd : “sudo apt-get install hostapd isc-dhcp-server”.
- Isc-dhcp-server: “sudo apt-get install hostapd isc-dhcp-server”.

5.5.1. Servidor DHCP

Con el rápido crecimiento de TCP/IP (*Trasmisión Control Protocol/Internet Protocol*), que es un método de transmisión para comunicarse en Internet, se necesitan algunas herramientas para administrar automáticamente algunas funciones gestionando redes TCP/IP. DHCP (*Dynamic Host Configuration Protocol*) es un conjunto de reglas para dar direcciones IP y opciones de configuración a ordenadores y estaciones de trabajo en una red. Una dirección IP es un número que identifica de forma única a un ordenador en la red, ya sea en una red corporativa o en Internet. Una dirección IP es análoga a un número de teléfono.

La dirección IP puede ser asignada de forma estática (manualmente) por el administrador o asignada de forma dinámica por un servidor central.

Funcionamiento de DHCP

DHCP funciona sobre un servidor central el cual asigna direcciones IP a otras máquinas de la red. Este protocolo puede entregar información IP en una LAN o entre varias VLAN. Esta tecnología reduce el trabajo de un administrador, que de otra manera tendría que visitar todos los ordenadores o estaciones de trabajo uno por uno. Para introducir la configuración IP consistente en IP, máscara, gateway, DNS, etc.

Un servidor DHSC (*DHCP Server*) es un equipo en una red que está corriendo un servicio DHCP. Dicho servicio se mantiene a la escucha de peticiones broadcast DHCP. Cuando una de estas peticiones es oída, el servidor responde con una dirección IP y opcionalmente con información adicional.

Modos en DHCP

Existen 3 modos en DHCP para poder asignar direcciones IP a otros equipos:

- Asignación manual: El administrador configura manualmente las direcciones IP del cliente en el servidor DCHP. Cuando la estación de trabajo del cliente pide una dirección IP, el servidor mira la dirección MAC y procede a asignar la que configuró el administrador.
- Asignación automática: Al cliente DHCP (ordenador, impresora, etc.) se le asigna una dirección IP cuando contacta por primera vez con el DHCP Server. En este método la IP es asignada de forma aleatoria y no es configurada de antemano.
- Asignación dinámica: El servidor DHCP asigna una dirección IP a un cliente de forma temporal. Digamos que es entregada al client Server que hace la petición por un espacio de tiempo. Cuando este tiempo acaba, la IP es revocada y la estación de trabajo ya no puede funcionar en la red hasta que no pida otra.

DHCP es un protocolo diseñado principalmente para ahorrar tiempo gestionando direcciones IP en una red grande. El servicio DHCP está activo en un servidor donde se centraliza la gestión de la direcciones IP de la red.

Instalación y configuración del servidor DHCP

En primer lugar instalaremos el servidor que nos permitirá crear el punto de acceso Wifi:

- Isc-dhcp-server: “sudo apt-get install hostapd isc-dhcp-server”.

DHCP es el protocolo que hace que automáticamente se nos asigne una dirección IP al conectarnos a una red. Para configurar-lo, hemos de seguir los siguientes pasos:

1. Abrir el fichero de configuración con el editor Nano : “sudo nano /etc/dhcp/dhcpd.conf”.
    - Buscar y comentar las líneas :

```
#option domain-name "example.org";
#option domain-name-servers ns1.example.org, ns2.example.org;
```

Figura 38:Líneas a comentar

- Buscar y descomentar la siguiente linea:

```
authoritative;
```

Figura 39:Líneas a descomentar

- Para acabar con los cambios en el fichero de configuración añadimos las siguientes líneas al final de este:

```
subnet 192.168.100.0 netmask 255.255.255.0 {
        range 192.168.100.10 192.168.100.50;
        option broadcast-address 192.168.100.255;
        option routers 192.168.100.1;
        default-lease-time 600;
        max-lease-time 7200;
        option domain-name "local";
        option domain-name-servers 8.8.8.8, 8.8.4.4;
}
```

Figura 40:Líneas a añadir

2. Ahora continuaremos la configuración en el archivo “isc-dhcp-server”:
   - Lo abrimos con Nano: “sudo nano /etc/default/isc-dhcp-server”
   - Buscamos la línea INTERFACES=”” y entre las comillas añadimos “wlan0“.
   - Guardamos y cerramos el archivo.

Configurar una IP estática en wlan0

Para tener siempre la misma IP hemos de acceder al fichero interfaces, para ello seguimos los siguientes pasos:

1. Antes de abrir el fichero desactivamos la interfaz wlan0 con el siguiente comando: “sudo ifdown wlan0”
2. Abrimos el fichero de configuración: “sudo nano /etc/network/interfaces”
3. Cambiamos el archivo, este debe quedar así:

```
auto lo

iface lo inet loopback
iface eth0 inet dhcp


auto wlan0
allow-hotplug wlan0
iface wlan0 inet static
        address 192.168.100.1
        netmask 255.255.255.0
```

Figura 41:Archivo interfaces

4. Guardamos y cerramos el archivo. Ahora solo queda asignar la IP a la interfaz wlan0 y habilitarla con el siguiente comando: “sudo ifconfig wlan0 192.168.100.1”

5.5.2. HOSTAPD

HOSTAPD es un excelente software que permite convertir un Linux en un punto de acceso wifi totalmente configurable. La configuración escogida es WPA2 por seguridad, para evitar que otras personas ajenas tengan acceso a nuestra red WI-FI del emulador.

Las siglas WPA significan; Wlan Protected Access. Es un sistema para proteger las redes inalámbricas (Wi-Fi); creado para corregir las deficiencias del sistema previo, Wired Equivalent Privacy (WEP). WPA implementa la mayoría del estándar IEEE 802.11i, y fue creado como una medida intermedia para ocupar el lugar de WEP mientras 802.11i era finalizado. WPA adopta la autentificación de usuarios mediante el uso de un servidor, donde se almacenan las credenciales y contraseñas de los usuarios de la red. Para no obligar al uso de tal servidor para el despliegue de redes, WPA permite la autentificación mediante una clave precompartida, que de un modo similar al WEP, requiere introducir la misma clave en todos los equipos de la red. Una vez finalizado el nuevo estándar 802.11ie se creo el WPA2 basado en este. La configuración del servidor se realiza con los siguiente parámetros dentro de hostapd.conf:

Características básicas

- driver: driver a usar por el demonio hostapd para comunicarse con el kernel. Normalmente es nl80211, que soporta muchas tarjetas de red, pero hay otros más antiguos para casos específicos: hostap, madwifi y prism54.
- ssid: SSID de la red wifi que se va a crear desde el punto de acceso.
- interface: nombre de la tarjeta de red wifi (normalmente wlanX o athX).
- channel: canal usado la red wifi creada. Debe estar entre 1 y 11, pero recordemos los canales se superponen físicamente en el espacio de frecuencias , y que los canales óptimos para que no existan conflictos son 1, 6 y 11. Versiones modernas de hostapd permite el valor "auto", de tal forma que se busca el canal menos saturado dentro del entorno donde esté la red wifi.
- hw_mode: modo de red usado por la red wifi. Puede ser a (11mbps), b (22mbps) y g (54mbps). El modo n (104mbps) se configura aparte, con otro parámetro.

Características de seguridad

- ignore_broadcast_ssid: puesto a 1 oculta la difusión del SSID de la red wifi. De esta manera ningún dispositivo normal podrá verla en su lista de redes wifi y solo podrá conectarse a ella dando su nombre exacto. Es un sistema de seguridad rudimentario y fácil de sortear usando una herramienta de auditoría wifi que haga un sniffing de las redes en el espacio radioeléctrico, pero normalmente es bastante efectivo ante intrusos de bajo nivel.

- macaddr_acl: con valor 1 activa el filtro MAC en el punto de acceso, puesto a 0 lo desactiva.

- auth_algs=1/2/3: En el caso de WPA ha de estar a uno, cas características de configuración de este campo son:.
    1. Open Auth. Usar el método de autenticación "Autenticación Abierta"
    2. Shared Keys. Usar el método de autenticación "Claves compartidas".
    3. Open Auth y Shared Keys. Se usa cualquiera de los dos métodos.

- Wpa = 0/1/2/3 . El parámetro wpa puede ser: 0 (sin wpa), 1 (usar wpa), 2 (usar wpa2) y 3 (usar wpa y wpa2)

- wpa_passphrase = “ “. Se indica la contraseña de la red, esta ha de contener entre 8 y 63 carácteres.

- wpa_key_mgmt = WPA-PSK/WPA-EAP/WPA-PSK WPA-EAP
    - WPA-PSK necesita wpa_psk o wpa_passphrase
    - WPA-EAP es para servidores EAP y Radius, que son un sistema que restringe el acceso por usuario y contraseña.

- wpa_pairwise=TKIP/CCMP . Algoritmo de cifrado usado.

- rsn_paiwise=CCMP . Algoritmo de cifrado usado.

Configuración Hostapd

Para esta configuración hemos de crear un nuevo archivo “hostapd.conf”, en el se definen todas las configuraciones de la Wlan0 (SSID, contraseña, tipo de seguridad, etc). En nuestro caso solo configuro la interfaz, el driver, el SSID, el modo de hardware y el canal, el resto de configuraciones no he querido configurarlas porque el dispositivo solo será un emulador de rutas, con lo que el acceso a terceros será muy difícil por su bajo alcance y por no ofrecer salida a Internet.

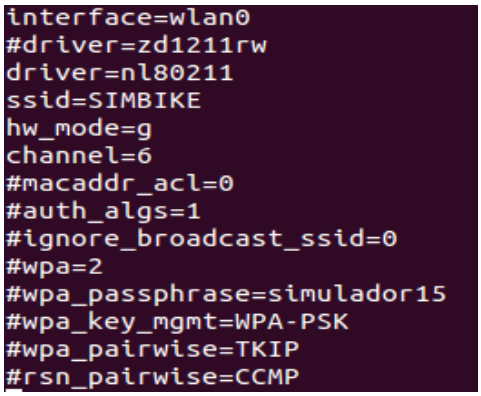


Figura 42:Líneas a descomentar

Un aspecto importante durante la configuración el el driver de la antena wifi, como se ve en este archivo el driver “zd1211rw” está comentado, éste es el driver aconsejado por el fabricante, pero probándolo el dispositivo wifi no funcionaba. Para conseguir hacer funcionar la antena wifi se ha usado un driver genérico “nl80211” compatible. Se tiene que vigilar de no dejar líneas extras ni espacios después de cada línea, podría causar problemas.

Por último modificamos el archivo “/etc/default/hostapd” para decirle a Raspberry pi donde encontrar el archivo que acabamos de crear. Descomentamos y añadimos la ruta en la siguiente línea:

```
DAEMON_CONF="/etc/hostapd/hostapd.conf"
```

Figura 43:Líneas a descomentar y modificar

Activación de nuestro punto de acceso

Para activar nuestro punto de acceso simplemente hemos de introducir los siguientes comandos:

```
sudo service hostapd start
sudo service isc-dhcp-server start
```

Figura 44:Líneas a descomentar y modificar

Para que el que nuestro punto de acceso se active cada vez que se inicia nuestro emulador tenemos que habilitar un “daemos” de la siguiente manera:

```
sudo update-rc.d hostapd enable
sudo update-rc.d isc-dhcp-server enable
```

Figura 45:Líneas a descomentar y modificar

Ahora ya tenemos nuestro emulador con conexión Wifi abierta al usuario, la cual se inicia cada vez que se inicia el dispositivo emulador.

5.6. Componentes e instalación

Los componentes usados para la elaboración de emulador son:.

Raspberry pi

Raspberry Pi es un minicomputador de bajo coste diseñado para el ámbito académico con el objetivo de cubrir las funciones que un ordenador personal puede dar a un precio muy competitivo. Para realizar el emulador el sistema operativo elegido es RaspBbian, este sistema operativo es libre basado en Debian y optimizado para el hardware Raspberry pi, este sistema operativo es el indicado para usar la Raspberry pi como servidor, como es este caso.

Figura 46: Raspberry pi B

Las funciones de este hardware son:

- Contener el servidor web que servirá al usuario la aplicación web. El servidor web instalado en la Raspberry es Apache 2.
- Funcionamiento como un punto de acceso, en concreto un punto de acceso wifi. Mediante el canal wifi que creara la Raspberry el usuario podrá conectarse a la aplicación web.
- Control de un servomotor mediante GPIO (General Purpose Input/Output), es un puerto que sirve a la Raspberry Pi para comunicarse con dispositivos externos. El puerto GPIO está formado por 26 pins los cuales se pueden configurar como entradas o salidas digitales. También incorpora pins de masa y alimentación de 5V y 3,3V, y pins de comunicación Serial, I2C y SPI pre-configurados. Estos pins trabajan a un voltaje de 3,3V y, al contrario que un Arduino, los pins GPIO de la Raspberry Pi no tienen ninguna protección de circuitería, por lo que hay que ir con cuidado a la hora de conectar dispositivos a estos pins.

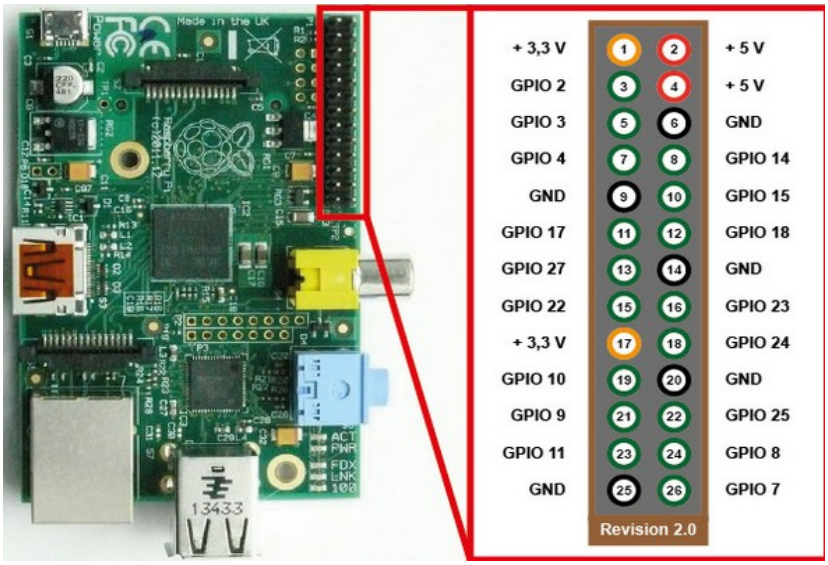


Figura 47:Servomotor SG90

Los pines se pueden referenciar por su número de pin de la placa (modo BOARD) o por el numero de GPIO (modo BCM).

Servomotor

El servomotor es un motor de corriente continua que permite controlar su posición mediante un eje. El elegido para el emulador es el modelo SG90 de la marca TowerPro. La posición del servo oscila entre 0º y 180º. El cableado del servomotor se compone de tres cables: marrón para el negativo, rojo el positivo y naranja para el señal de entrada de control.

Figura 48:Servomotor SG90

El servomotor es controlado por el cable naranja, desde el pin GPIO 9 de la Raspberry le enviamos un señal con un período de veinte mili-segundos y un pulso positivo de entre uno y dos mili-segundos en función a la posición que queremos. Con un pulso de 1ms el servo esta en la posición 0º y con 2ms a 180º de la posición inicial.

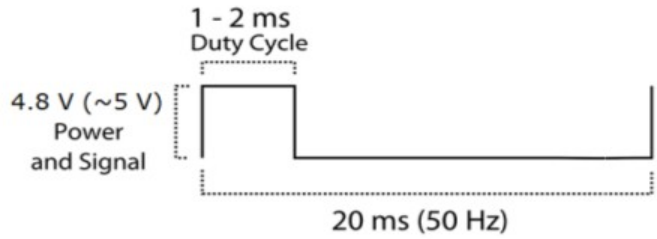


Figura 49:Período de trabajo del servomotor

En el montaje del servomotor se ha diseñado un soporte para atornillar el servomotor, este soporte a la vez hace de tensor con un regulador.

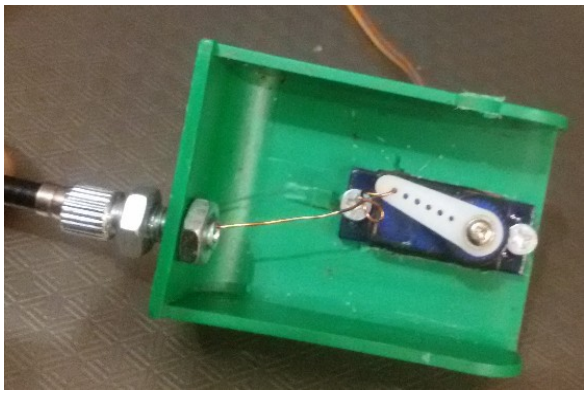


Figura 50:Motaje del servomotr

Antena wifi

Con el adaptador Wifi CT-wn4322Z compatible con Raspberry pi creamos una red Wifi en el emulador y así podremos conectarnos con cualquier dispositivo. El modelo es totalmente compatible con Raspbian. Éste modelo es económico y compatible con el estándar 802.11b/g/n.

Figura 51:Antena wifi ct-wn4322Z

Rodillo bicicleta

El emulador tiene también un componente mecánico, unos rodillos de entrenamiento. Valorando varias opciones se ha elegido el modelo “Novo Force Elastogel” de la marca “Elite”. Las características de este dispositivo de entrenamiento son:

- Se adapta a ruedas con diámetro comprendido entre 24" y 29".

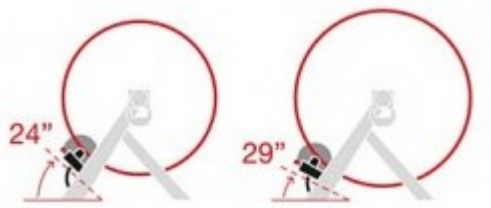


Figura 52:Tamaños de rueda

- El material de contacto entre el rodillo y el neumático es elastogel, que reduce el nivel de ruido en un 50%, disminuye el desgaste del neumático en un 20% y aumenta la adherencia respeto a los antiguos materiales de contacto.

Figura 53:Elastogel

- El bastidor tiene mucha superficie de apoyo, una baja elevación de la bici y una gran solidez estructural, estas características garantiza una buena estabilidad al rodillo incluso con altas velocidades o con el máximo esfuerzo. El bastidor en el proyecto nos ha servido como soporte para el hardware del emulador

Figura 54:Montaje

Figura 55:Bastidor

- El sistema de soporte de la unidad permite una idónea y constante presión entre rodillo y neumático, además el anclaje se realiza con un sistema simple de presión.

Figura 56:Fijación de eje

- El selector de nivel de esfuerzo es mecánico mediante la tensión de un cable de acero, el cambio entre niveles es con un sector rotativo, esta característica facilita el diseño de implementación con un servo motor.

Figura 57:Selector de resistencia

Problemática con el montaje

El selector del rodillo ejerce mucha fuerza sobre el cable que acciona. La fuerza es generada por un muelle interno del rodillo y la fuerza magnética que ejerce un imán para generar resistencia de giro, las fuerzas que ejercen son demasiado fuertes para el servomotor, para solucionar este problema se ha sustituido el muelle original por otro sistema de muelles con mucha menos fuerza (garantizando el buen funcionamiento)

Figura 58:Variaciones del mecanismo

6. Resultado

A continuación se exponen los resultados obtenidos de la realización del proyecto:

- Se ha creado una aplicación que cumple con todas las expectativas a pesar de las dificultades por la falta de configuraciones que ofrece App Inventor 2. Esta aplicación es capaz de registrar los datos de ubicación a la vez que los de posicionamiento, también se sincroniza a la perfección con la cámara de deporte
- Los sensores de ubicación no son todo lo precisos que se desearía y el valor de la altitud no es nada fiable.
- La aplicación gestiona la base de datos de las rutas y sus informaciones de manera rápida, pero no tiene posibilidad a borrar archivos que ha creado, dejando así basura en el dispositivo.
- La grabación del entrenamiento se ha de realizar con la pantalla siempre activa, sino el sensor que nos mide el angulo deja de registrar valores. El usuario ha de configurar su dispositivo para que quedarse siempre activo.
- No se ha podido ofrecer una forma intuitiva para la compartir de los archivos de información (“nombre_ruta.txt” y vídeo de la ruta). Se ha intentado hacer desde la misma aplicación Android, pero los recursos para compartir información no permitían la compartir de archivos. Esto se ha solucionado con una parte del la aplicación web dedicada a la subida de información, pero el usuario ha de unificar los dos archivos en un mismo dispositivo.
- El emulador ofrece una conexión Wifi independiente con seguridad WPA2.
- La aplicación Web tiene una interfaz gráfica de fácil comprensión, además ofrece una gestión de las rutas completa (visualización, borrado y emulación).
- En la emulación de un entrenamiento por primera vez puede haber problemas con la sincronización entre el vídeo y el programa de control del servo, esto es generado por el tiempo de carga del vídeo. Cuando el vídeo ya se ha reproducido con anterioridad, la reproducción de este no se ve afectado.

Posibles mejoras

Existen infinidad de mejoras, este es un proyecto ampliable en muchos aspectos, a continuación vamos a proponer algunas de ellas:

- Aplicación Android, sin las limitaciones de App Inventor 2, crear la aplicación con Android studio nos daría la posibilidad los puntos débiles existentes anteriormente mencionados.
- Arduino, la incorporación de un Arduino al emulador sería provechoso para poder incorporar más sensores durante la emulación para conocer datos de esta (ritmo de pedaleo, velocidad, ritmo cardíaco, etc.)

- Crear una retroalimentación en el emulación, es decir, el proceso de visualización de vídeo reproduzca más o menos rápida en función a un sensor de velocidad instalado en el dispositivo de entrenamiento. Esta característica nos daría más realismo al entrenamiento realizado.
- Mostrar información de forma gráfica, durante la realización de la aplicación web se ha trabajado en un módulo para la creación de gráficos. Este módulo creado genera un gráfico para cada cambió en el nivel de esfuerzo del dispositivo de entrenamiento, así el usuario puede ver en que nivel de esfuerzo esta trabajando. También sería buena opción el uso de otros gráficos, como el de la ascensión que se esta realizando en valores de altitud.

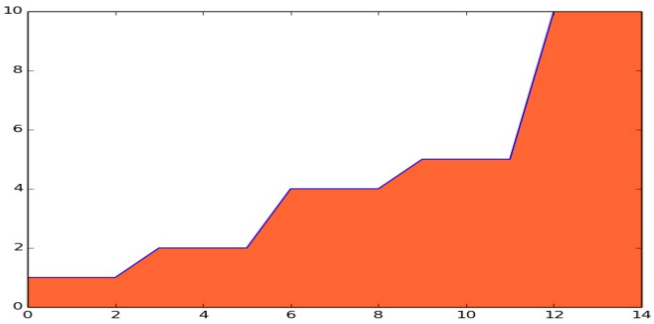


Figura 59:Variaciones del mecanismo

Este gráfico esta creado con el modulo “gráfico.py” adjuntado en la documentación. Este módulo la intención era que generase un gráfico del nivel de esfuerzo en que se encuentra el dispositivo emulador cada tres segundos, el mismo tiempo que el sistema se mantiene en un mismo nivel, pero el tiempo de ejecución de este demasiado elevado (una media de 6 segundos) y no ha sido posible sincronizarlo con la emulación, por lo tanto no muestra datos importantes.

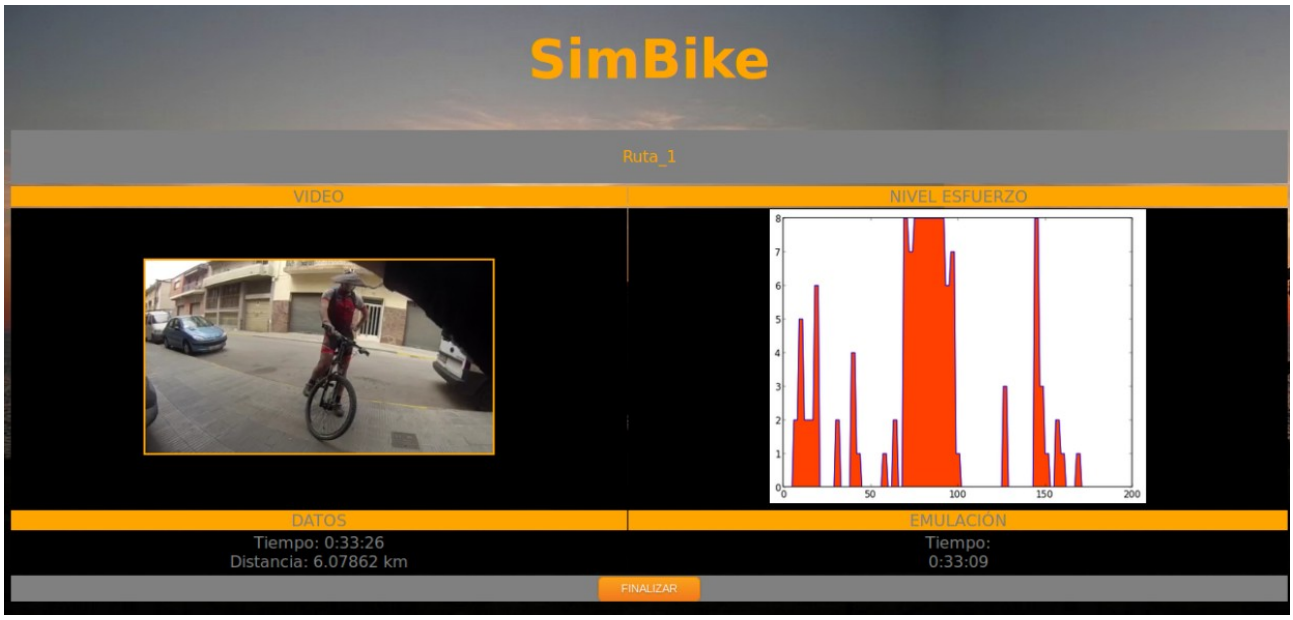


Figura 60: Pruebas con el módulo “grafico.py”

7. Conclusión

El resultado de la realización de este proyecto de final de grado es un emulador de entrenamiento de bicicleta muy completo, con una interfaz gráfica tanto en la aplicación Android como en la aplicación Web de gran calidad y con funcionalidades muy completas.

En la aplicación Android se ha de destacar las gran cantidad de dificultades que me he encontrado causadas por la baja configuración de la que se dispone desde App Inventor 2. Otro aspecto que ha complicado mucho la elaboración de la aplicación Android ha sido los grandes códigos de bloques que se han necesitado para realizar pequeñas cosas, esta gran extensión de bloques también ha dificultado el funcionamiento de App Inventor, provocando continuos bloques de la la aplicación web de desarrollo e impidiendo incluso trabajar.

En el emulador de entrenamiento se ha logrado tener una conexión Wifi independiente que aporta gran valor al dispositivo, este ofrece una aplicación web muy completa y con un funcionamiento y sincronización muy buenos, ofreciendo un producto con muy buen funcionamiento.

8. Bibliografía

1. Control de cámara GoPro. Web. https://github.com/KonradIT/goprowifihack/blob/master/WiFi-Commands.mkdn
2. Tutorial servidor web. Web. https://geekytheory.com/tutorial-raspberry-pi-crear-servidor-web/
3. Configuración Raspberry pi. Web. http://raspberryparatorpes.net/empezando/raspi-config-2014-configuar-raspbian-paso-a-paso/
4. Manual php. Web.http://php.net/
5. Solución máximo tamaño archivos php. Web. http://stackoverflow.com/questions/6327965/html-upload-max-file-size-does-not-appear-to-work
6. Vídeo html5. Web. http://www.html5rocks.com/es/tutorials/video/basics/
7. Recursos Html. Web. http://www.w3schools.com/html/html5_intro.asp
8. Turorial Raspberry pi punto de acceso. Web. https://geekytheory.com/tutorial-rasbperry-pi-como-crear-un-punto-de-acceso-wifi/
9. Tutorial Raspberry pi GPIO con servomotor. Web. http://fpaez.com/controlar-un-servomotor-con-raspberry-pi/
10. Arduino schematic. PDF. https://www.arduino.cc/en/uploads/Main/arduino-uno-schematic.pdf
11. Arduino datasheet, PDF. http://www.atmel.com/Images/Atmel-8271-8-bit-AVR-Microcontroller-ATmega48A-48PA-88A-88PA-168A-168PA-328-328P_datasheet_Complete.pdf

9. Anexo

9.1. Contenido de la entrega

En la entrega en formato DVD podemos encontrar dos carpetas:

- Directorio “Aplicación Web”. Contiene todos los archivos del servidor tal cual los encontramos en el servidor creado, misma distribución.
- Directorio “Aplicación Android”. Contiene el archivo apk para la instalación de la aplicación en cualquier dispositivo Android y un fichero con extensión “.aia”, correspondiendo al proyecto de App Inventor. Con este último archivo podemos abrir el proyecto de App Inventor desde su página Web y poder visualizar todo el diseño gráfico y de bloques de la aplicación.
- Directorio “Ampliaciones”, en el es se incluyen el modulo de creación de gráficos y un ejemplo completo de vídeo de html5.

9.1. Ejemplo completo vídeo html

El reproductor de vídeo html es muy potente y con gran posibilidad de configuración. Esta etiqueta es una de las grandes novedades de la quinta revisión del código html, html5. A continuación se muestra un reproductor de vídeo y en la entrega de la práctica se entrega el código del mismo:

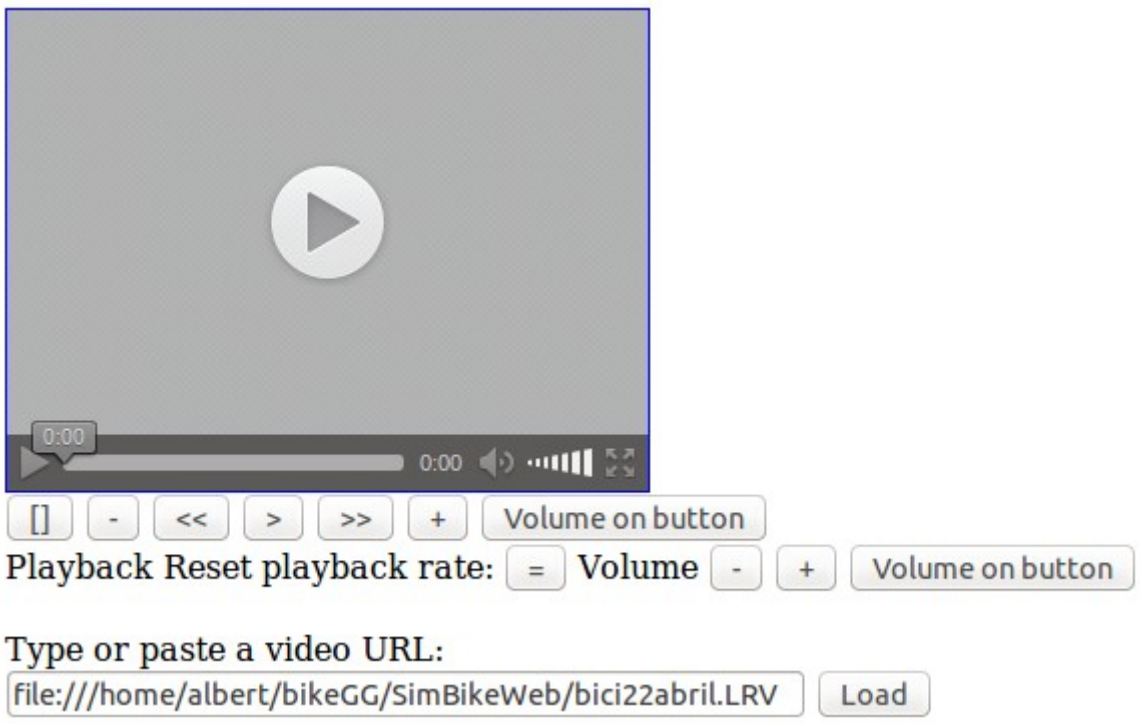


Figura 61: Ejemplo vídeo html